Resilience evaluation with regard to accidental and malicious threats

Mohamed Kaâniche

mohamed.kaaniche@laas.fr

Summer School

Resilience in Computing Systems and Information Infrastructures
- From Concepts to Practice -

24-28 September 2007, Porquerolles, France

Scope

- Dependability and Security
  - Attributes
    - Availability
    - Reliability
    - Safety
    - Confidentiality
    - Integrity
    - Maintainability
    - (Availability, Confidentiality, Integrity) → Security
  - Threats
    - Faults
    - Errors
    - Failures
    - accidental + malicious
  - Means
    - Fault Prevention
    - Fault Tolerance
    - Fault Removal
    - Fault Forecasting → resilience evaluation

Outline

- Introduction:
  - ordinal and quantitative evaluation
- Definitions of quantitative measures
- Probabilistic evaluation methods
  - Combinatorial models: Reliability diagrams, Fault trees
  - State-based models: Markov chains
- Experimental measurements
- Evaluation with regard to malicious threats
- Conclusion

Resilience evaluation

- Estimate the present number, the future incidence and the likely consequences of faults
- Assess the level of confidence to be placed in the target systems with regards to their ability to meet specified objectives
- Support engineering and design decisions
  - comparative evaluation of candidate architectures
  - prediction of the level of resilience to be achieved in operation
  - reliability, resource and cost allocation based on quantified predictions

Two types of evaluation

Qualitative or "ordinal"

identify, classify, and rank the failure modes or the event combinations (component failures or environmental conditions) that would lead to system failures

Quantitative or "probabilistic"

evaluate in terms of probabilities the extent to which some of the attributes are satisfied

attributes ⤞ measures

Evaluation methods

Ordinal evaluation		Probabilistic evaluation
→	FMECA	
→	Reliability block diagrams	←
→	Faut Trees	←
	State-based models	
	Markov chains	←
→	Petri nets	←

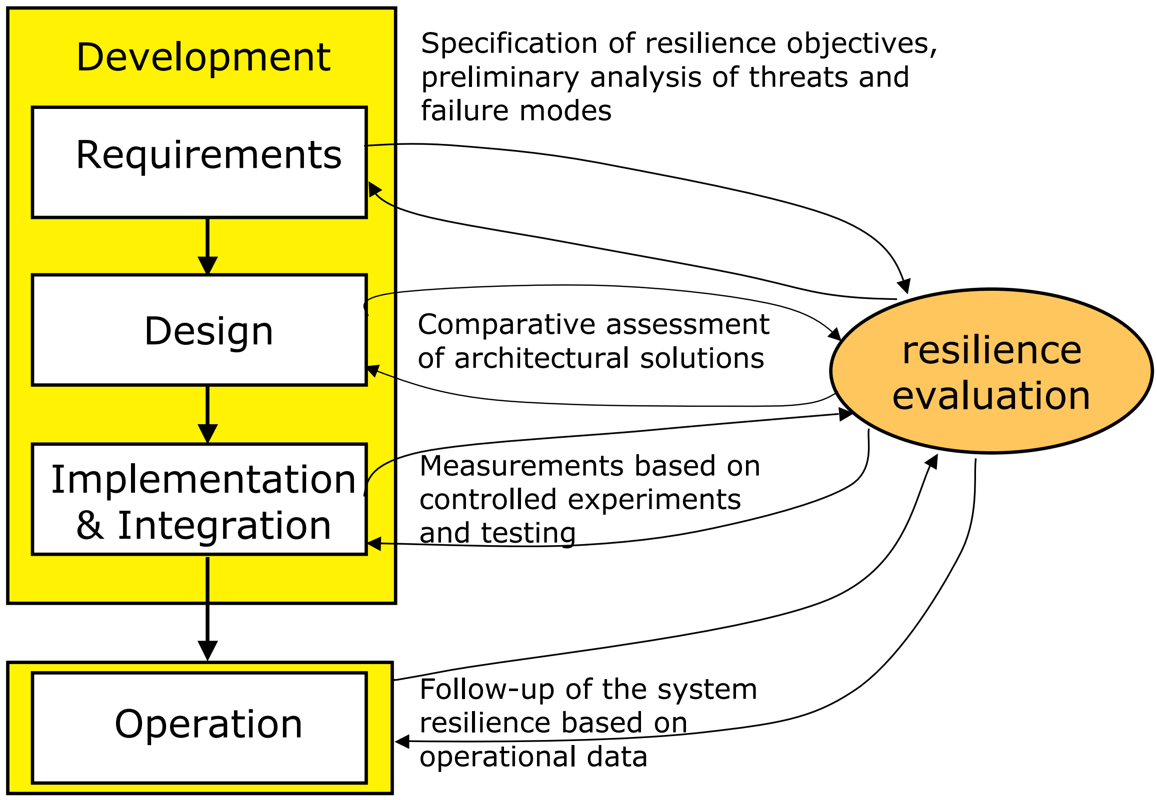
Modelling (analytical models, simulation)

relevant parameters to measure

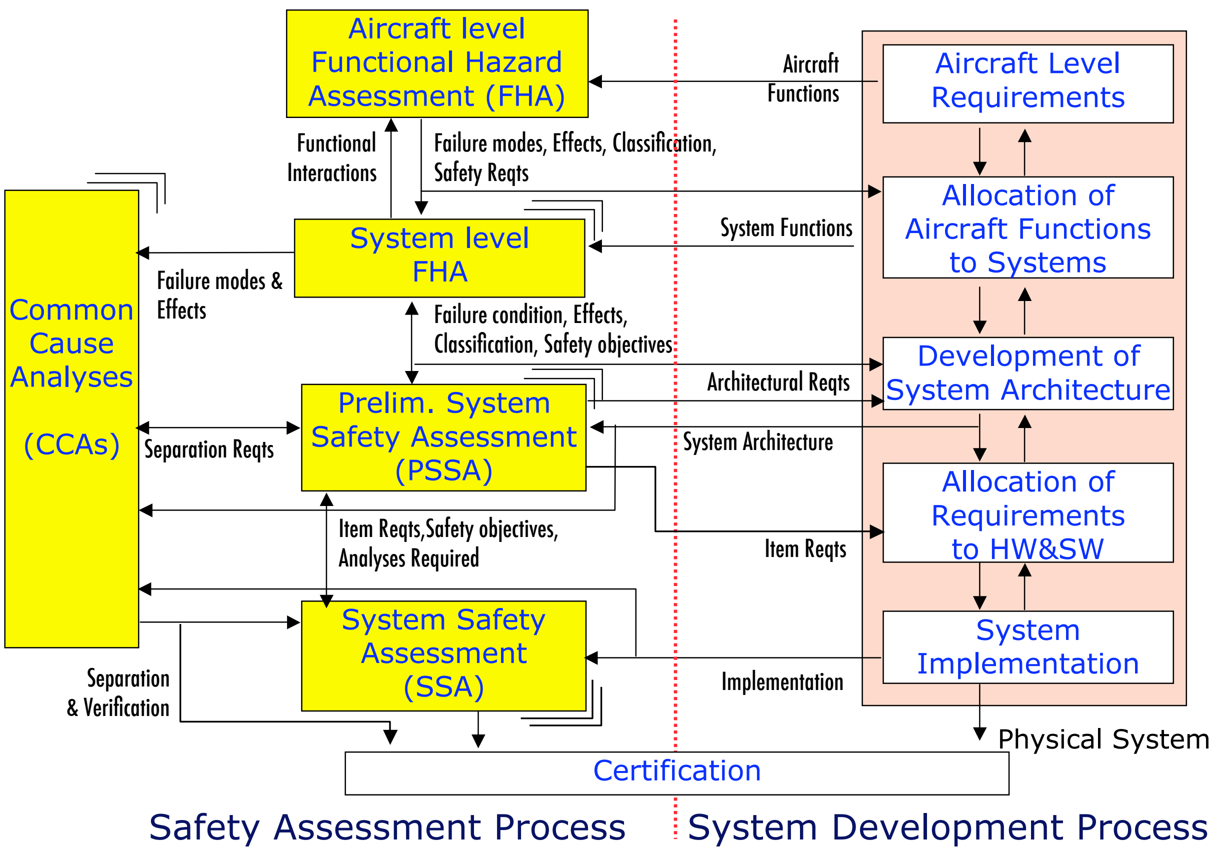
parameter estimation
model validation

Measurements (controlled experiments, data from operation)

Place in the life cycle process



Avionics: ARP 4754 Standard



FMECA

Failure Modes, Effects, and Criticality Analysis

- Initially used for Hardware, then extended to software (SEEA: *Software Error Effect Analysis*)
- What can FMECA be used for?
  - Identify for each component, or function, .. potential failure modes and their consequences on the system
    - failure mode = the way a failure manifests itself
  - Assess the criticality of each failure mode
    - failures prioritized according to how serious their consequences are and how frequently they occur
  - Identify possible means to prevent or reduce the effects of each failure mode
  - Define validation tests to analyze such failure modes

Generic failure modes (IEC 812-1985)

1. Structural failure (rupture)
2. Physical binding or jamming
3. Vibration
4. Fails to remain in position
5. Fails to open
6. Fails to open
7. Fails open
8. Fails closed
9. Internal leakage
10. External leakage
11. Fails out of tolerance (high)
12. Fails out of tolerance (low)
13. Inadvertent operation
14. Intermittent operation
15. Erratic operation
16. Erroneous indication
17. Restricted flow
18. False actuation
19. Fails to stop
20. Fails to start
21. Fails to switch
22. Premature operation
23. Delayed operation
24. Erroneous input (increased)
25. Erroneous input (decreased)
26. Erroneous output (increased)
27. Erroneous output (decreased)
28. Loss of input
29. Loss of output
30. Shorted (electrical)
31. Open (electrical)
32. Leakage (electrical)
33. Other unique failure conditions as applicable to the system characteristics, requirements and operational constraints

Criticality {severity, frequency}

Frequency	Severity			
	Catastrophic	Critical	Marginal	Negligeable
Frequent	Class I			
Probable				
Occasional		Class II		
Remote			Class III	
Improbable				
Incredible				Class IV

Example: IEC- 61508-5 standard

Class I	Untolerable risk. Risk reduction measures are required
Class II	Undesirable risk, tolerable only if risk reduction is impractical or if costs are disproportionate to the improvment gained
Class III	Tolerable risk if the cost of risk reduction would exceed the improvment gained
Class IV	Negligible risk

FMECA steps

- Breakdown the system into components
- Identify the functional structure and how the components contribute to functions
- Define failure modes of each component, their causes, effects and severities
  - Local effect: on the system element under study
  - Global effect: on the highest considered system level
- Enumerate possible means to detect and isolate the failures
- Identify mitigation actions to prevent of reduce the effects of failure at the design level or in operation

FMECA Worksheet

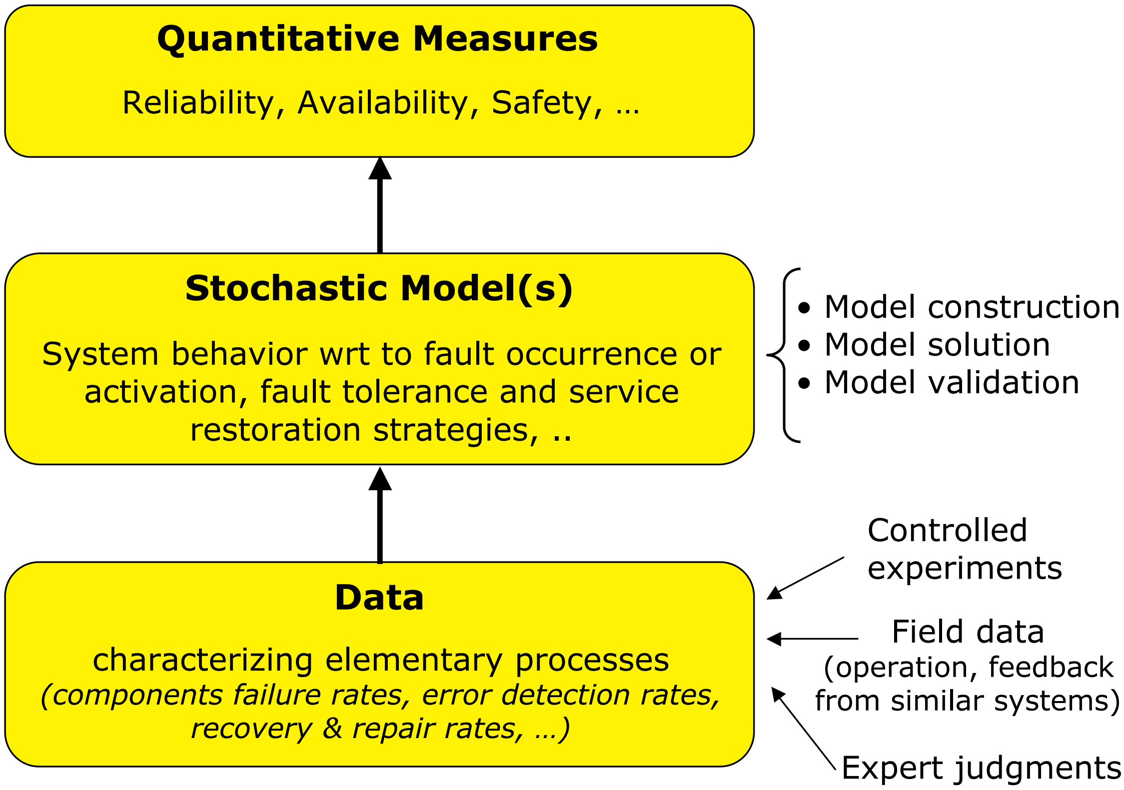
Description of unit			Description of failure		Failure effect	
Ref. n°	function	operational mode	failure mode	failure cause	local	global

Detection & mitigation				
detection means	corrective actions	*Probability of occurrence*	*Criticality level*	Comments

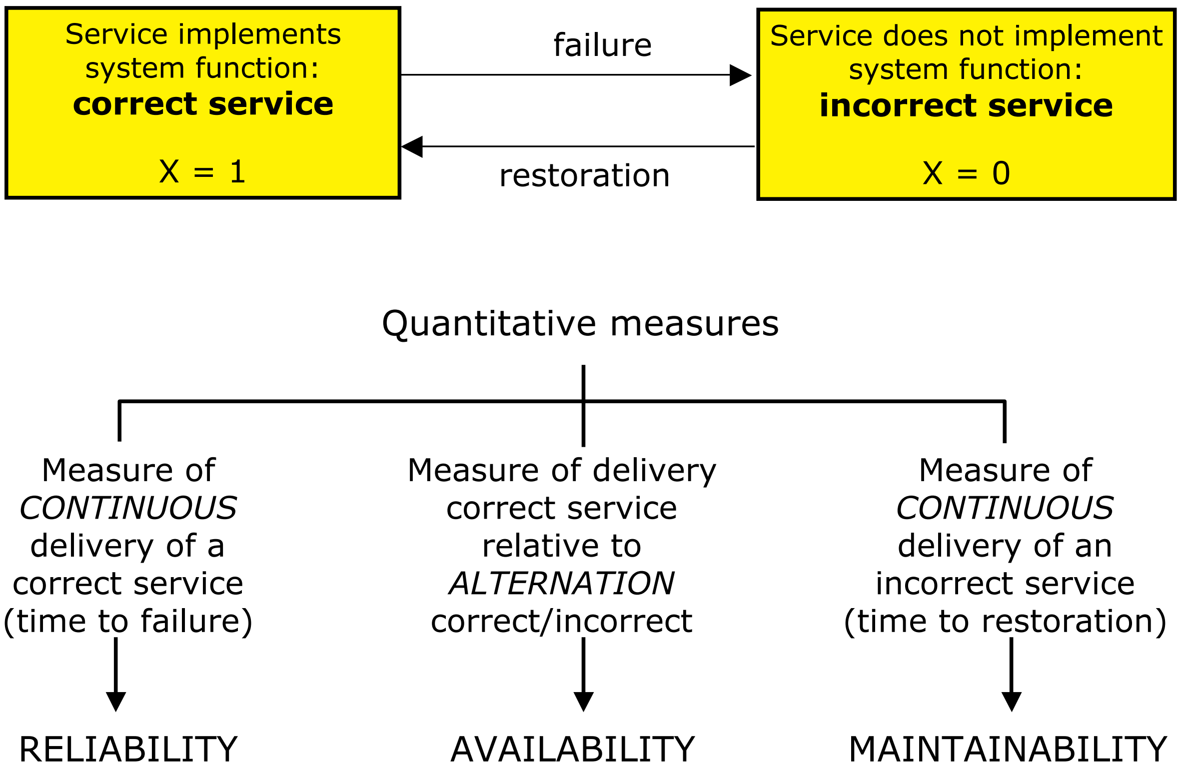
FMECA pros and cons

- Pros
  - applicable at the early design stage
  - detailed information about failure modes end their effects, during the various stages of the system development
  - contribution to the prevention of design faults and to the definition of fault tolerance requirements and needs
  - useful inputs for validation testing
  - results can be used to aid in failure analysis and maintenance during operation
- Cons
  - not suitable for multiple failures
  - may be tedious for complex systems .. but .. necessary

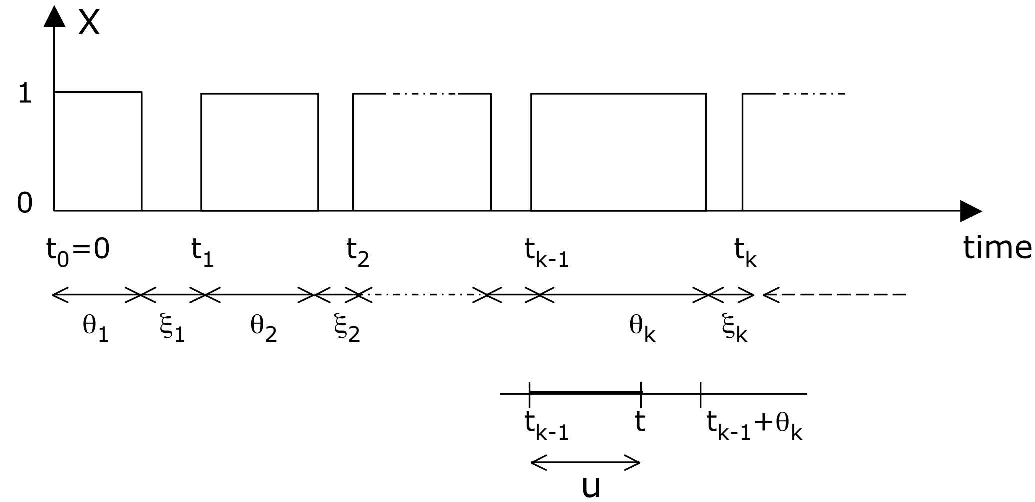
Probabilistic evaluation



Dependability measures



Dependability measures



Reliability: $R_k(u)=\text{Prob. }\{\theta_k > u\} = \text{Prob.}\{X(\tau)=1 \;\; \forall\, \tau \in [t_{k-1}, t_{k-1}+u]\}$

Availability: $A(t) = \text{Prob. }\{X(t) = 1\} = E\,\{X(t)\}$

Maintainability: $M_k(u) = \text{Prob. }\{\xi_k \leq u\}$

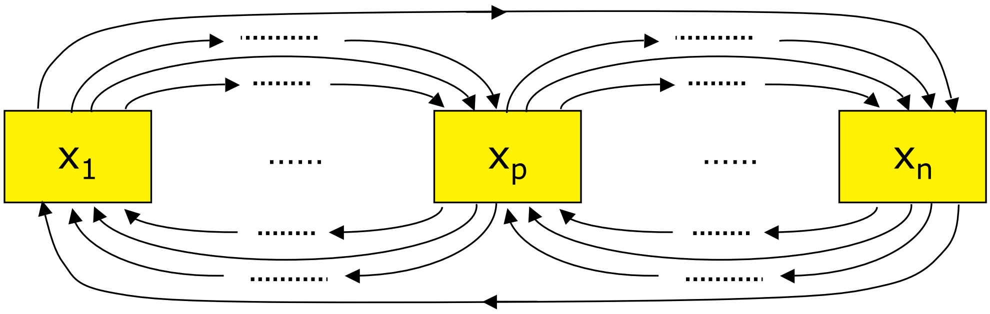
Multi-performing systems

- More than two service delivery modes
  - Correct service ➠ progressive performance degradation
  - Incorrect service ➠ failure consequences
- $X = \{x_1, x_2, \ldots x_n\}$
  - x_k: service delivery modes (accomplishment levels)
  - two extreme cases
    - 1 correct service mode — several incorrect service modes
    - Several correct service modes — 1 incorrect service mode
  - x_k are usually ordered, order induced by
    - performance levels: $perf(x_1) > perf(x_2) > \ldots > perf(x_n)$
    - criticality levels: $crit(x_1) > crit(x_2) > \ldots > cri(x_n)$

$$\Downarrow$$

$$x_1 > x_2 > \ldots > x_n$$

Multi-performing systems: measures



- Reliability-like measures:
  continuous delivery of service according to modes $\{x_1, \ldots x_p\}$
  (time to service delivery in modes $\{x_{p+1}, \ldots x_n\}$)

$$R_p(t) = \text{Prob.}\left\{X(\tau) \in \{x_1, \ldots, x_p\}\ \forall\, \tau \in [0, t]\right\}$$

- Availability-like measures:
  service delivery according to $\{x_1, \ldots x_p\}$ relative to alternation between modes $\{x_1, \ldots x_n\}$

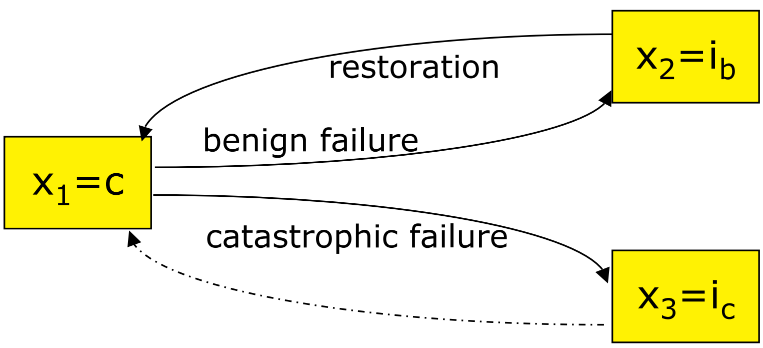
$$A_p(t) = \text{Prob.}\left\{X(t) \in \{x_1, \ldots, x_p\}\right\}$$

Particular cases

- 1 correct service mode: $x_1 = c$
- 2 incorrect service modes with very different severity levels
  - Benign incorrect service: $x_2 = i_b$
  - Catastrophic benign service: $x_2 = i_c$

$R_1(t) = \text{Prob.}\left\{X(\tau) = c\ \forall\, \tau \in [0, t]\right\}$ ➠ Reliability

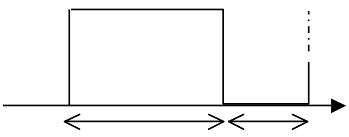
$R_2(t) = \text{Prob.}\left\{X(\tau) \in \{x_1, x_2\}\ \forall\, \tau \in [0, t]\right\}$ ➠ Safety



MTTF, MTTR, MUT, MDT, MTBF

- MTTF: mean time to failure

  $MTTF_k = E\{\theta_k\}$



- MTTR: mean time to restoration (repair)

  $MTTR_k = E\{\xi_k\}$

- MUT: mean up time (correct service delivery cycle)

$$MUT = \lim_{k=\infty} \frac{E\{\theta_1\} + E\{\theta_2\} + \ldots + E\{\theta_k\}}{k}$$

- MDT: mean down time (incorrect service delivery cycle)

$$MDT = \lim_{k=\infty} \frac{E\{\xi_1\} + E\{\xi_2\} + \ldots + E\{\xi_k\}}{k}$$

- MTBF: mean time between failures

  $MTBF = MUT + MDT \approx MUT$

Availability

- $A(t) = \text{Prob. } \{X(t) = 1\} = E\{X(t)\}$ $\qquad \overline{A(t)} = 1\text{-}A(t) = \text{Prob. } \{X(t) = 0\}$
- U(T): cumulated uptime ("correct service delivery time") in [0,T]

$$\frac{1}{T} E\{U(T)\} = \frac{1}{T} E\{\int_0^T X(t)dt\} = \frac{1}{T}\int_0^T E\{X(t)\}dt = \frac{1}{T}\int_0^T A(t)\,dt = A_{av}(T)$$

Average Availability in [0,T] = proportion of cumulated uptime in [0,T]

Availability	0.99	0.999	0.9999	0.99999	0.999999
Unavailability	0.01	0.001	0.0001	0.00001	0.000001
Downtime (min/year)	5256	525.6	52.56	5.256	0.5256

- Interval Availability: $A_I(t) = \frac{1}{t}\int_0^t A(x)\,dx$

- Steady-state Availability: $A = \lim_{t=\infty} A(t) = \lim_{T=\infty} A_{av}(T)$

$$A = \frac{MUT}{MUT+MDT} \qquad \overline{A} = 1\text{-}A = \frac{MDT}{MUT+MDT}$$

$\Leftarrow$ *stable reliability*

Stable reliability

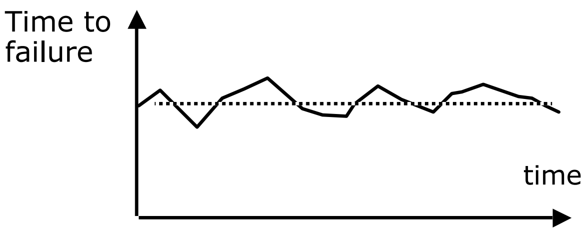
after restoration,
system identical to what it was
at previous restoration

Hardware → failed components replaced by identical non failed components

Software → restart

Identical stochastic distributions of times to failure θ_k:

$$Prob.(\theta_k < t) = Prob.(\theta_{k-1} < t)$$



Reliability growth

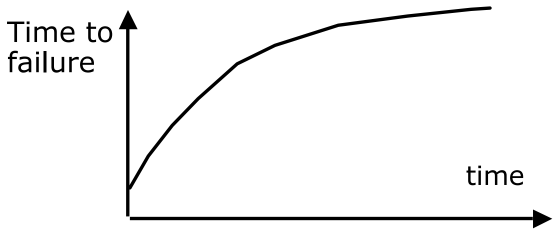
progressive removal of
residual development faults
(without introduction
of new faults)

↓

Reliability growth

Stochastic increase of times to failure θ_k:

$$Prob.(\theta_k < t) < Prob.(\theta_{k-1} < t)$$



Time to event occurrence characterization

θ : time to occurrence of a given event $\mathcal{E}$

name	symbol	definition	properties
Distribution function	$F(t)$	$Prob.(\theta \le t)$	monotonous increasing function: $F(0)=0$ $F(\infty)=1$
Complementary Distrib. function (survival funct.)	$\overline{F}(t)$	$Prob.(\theta > t)$	monotonous decreasing function: $\overline{F}(0)=1$ $\overline{F}(\infty)=0$
Probability density function	$f(t)$	$f(t).\Delta t = Prob.(t < \theta \le t+\Delta t)$ $f(t) = \frac{dF(t)}{dt} = \frac{-d\overline{F}(t)}{dt}$	$\int_0^\infty f(t).dt = 1$
hazard rate	$z(t)$	$z(t).\Delta t = Prob.(\theta \le t+\Delta t \mid \theta > t)$ $z(t) = \frac{1}{\overline{F}(t)} \frac{-d\overline{F}(t)}{dt}$	

Mean time to occurrence of event $\mathcal{E}$: $E(\theta) = \int_0^\infty t.f(t).dt = \int_0^\infty \overline{F}(t).dt$

Relationships between measures

	$F(t)$	$\overline{F}(t)$	$f(t)$	$z(t)$
$F(t)$	—	$1-\overline{F}(t)$	$\int_0^t f(x).dx$	$1-\exp.\int_0^t -z(x).dx$
$\overline{F}(t)$	$1-F(t)$	—	$\int_t^\infty f(x).dx$	$\exp.\int_0^t -z(x).dx$
$f(t)$	$\frac{dF(t)}{dt}$	$\frac{-d\overline{F}(t)}{dt}$	—	$z(t)\ \exp.\int_0^t -z(x).dx$
$z(t)$	$\frac{1}{1-F(t)}\ \frac{dF(t)}{dt}$	$\frac{1}{\overline{F}(t)}\ \frac{-d\overline{F}(t)}{dt}$	$\frac{f(t)}{\int_t^\infty f(x).dx}$	—

θ exponentially distributed with constant failure rate $z(t) = \lambda$

$F(t) = 1 - \exp.(-\lambda t)$ $\quad \overline{F}(t) = \exp.(-\lambda t)$ $\quad f(t) = \lambda \exp.(-\lambda t)$

$E(\theta) = 1/\lambda$

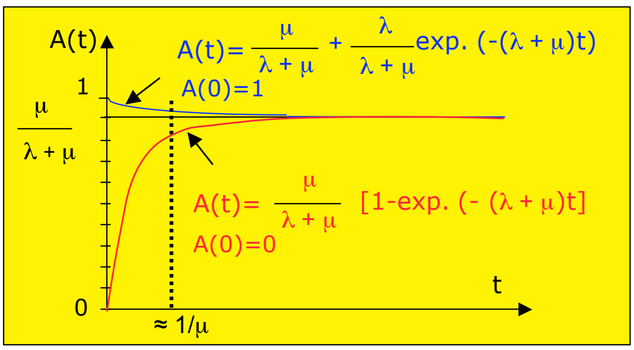
Single component system

- failure rate: $\lambda \Rightarrow$ MTTF $= 1/\lambda$
- restoration rate: $\mu \Rightarrow$ MTTR $=1/\mu$
- Reliability: $R(t) = \overline{F}(t) = \exp.(-\lambda t)$
- Availability: $A(t)$

A(t+dt)= Prob. (correct service at t AND no failure in [t, t+dt])
+ Prob. (incorrect service at t AND restoration in [t, t+dt])

$A(t+dt) = A(t)\,(1- \lambda dt) + (1-A(t))\,\mu dt$

$$\Rightarrow \frac{dA(t)}{dt} = \mu - (\lambda+\mu)A(t)$$



Multi-component systems modelling

- Model construction ➠ describe system behavior
  - Structure
  - components failures, fault tolerance/restoration strategies
- Model processing ➠ evaluate quantitative measures

Combinatorial models
(Reliability block diagrams, fault trees, reliability graphs, ..)

State-based models
(Markov chains, Stochastic Petri nets, non-Markovian models, ..)

Stochastically independent components

occurrence of event A *does not affect* event B
Prob. {A∩B} = Prob.{A} . Prob.{B}

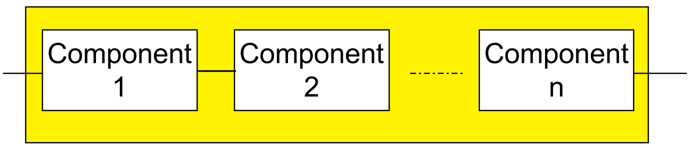
Stochastic dependencies

occurrence of event A affects event B
Prob. {A∩B} ≠ Prob.{A} . Prob.{B}

Reliability Block Diagrams

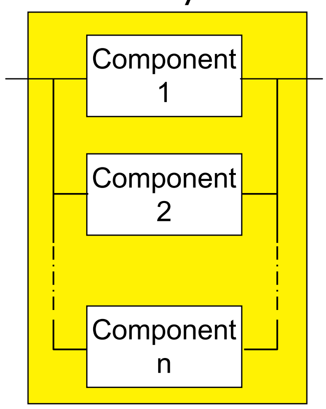
- Graph topology describing how components reliability affect system reliability
  - Each component represented as a block

SERIES
non-redundant systems



Components whose failure lead to system failure

PARALLEL
redundant systems



Components whose failure lead to system failure only in combination with others

GRAPH PATH ⇔ CORRECT SERVICE

Model processing

R_k: component k reliability, k = 1, …, n R: system reliability

- SERIES SYSTEMS

R= Prob. {system non failed}
R= Prob. {comp. 1 AND comp. 2 non failed AND comp. n non failed}

Stochastically independent components $\Rightarrow$ $R = \prod_{k=1..n} \{\text{comp. k non failed}\}$

$$R = \prod_{k=1..n} R_k$$

$R_k(t) = \exp. \int_0^t -\lambda(x).dx$ $R(t) = \exp.\{-\sum_{k=1..n} \lambda_k(x).dx\}$ $\Rightarrow \lambda(t) = -\sum_{k=1..n} \lambda_k(t)$

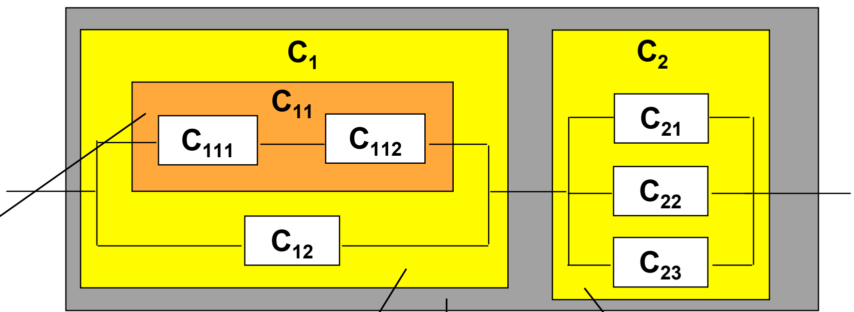
identical components with $\lambda_k(t) = \lambda$ $\Rightarrow$ MTTF= $1/(n\lambda)$

- PARALLEL SYSTEMS

System failed only when All components failed

$1-R = \prod_{k=1..n} \{1-R_k\}$ $R = 1-\prod_{k=1..n} \{1-R_k\}$

Parallel-Series systems



$$R_{11} = R_{111} . R_{112}$$

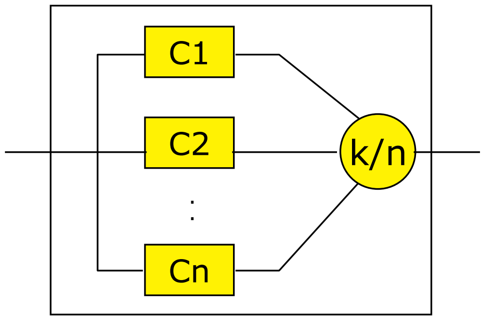
$$R_1 = 1-(1-R_{11}).(1-R_{12})$$

$$R_2 = 1-(1-R_{21}).(1-R_{22}).(1-R_{23})$$

$$\mathbf{R = R_1 . R_2}$$

“k-out of-n” systems with voter

- n components and a voter
- System non failed when less than k components failed



R_C: reliability of Component j
j=1,…n (identical comp.)

R_v: reliability of Voter

R: system reliability

$$R(t) = \left[\sum_{j=r}^{n} C_n^j \, [R_c(t)]^j \cdot [1 - R_c(t)]^{n-j} \right] \cdot R_v$$

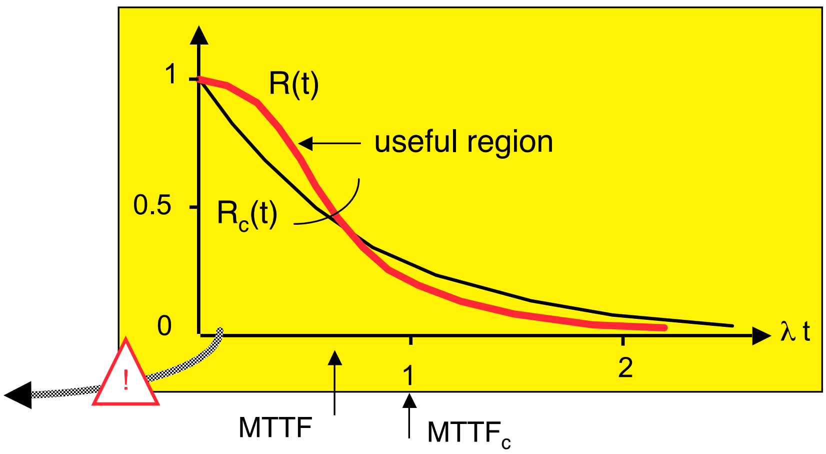
TMR systems

- “2-out of- 3” system with perfect voter

$$R(t) = 3 \, [R_c(t)]^2 - 2 \, [R_c(t)]^3$$

$$R_c(t) = \exp(-\lambda t) \quad \Rightarrow \quad R(t) = 3 \exp(-2\lambda t) - 2 \exp(-3\lambda t)$$

$$MTTF_c = 1 / \lambda \quad > \quad MTTF = 5 / 6 \lambda$$



Be careful when using
MTTF to characterize
dependability

Availability evaluation

- The same approach can be applied provided that the components are stochastically independent with respect to *failures* AND *restorations* ⇒ 1 repairman per component

A_k: component k availability, k=1, …, n

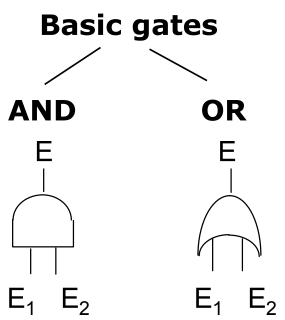
A: system availability

Series systems: $A = \prod_{k=1..n} A_k$

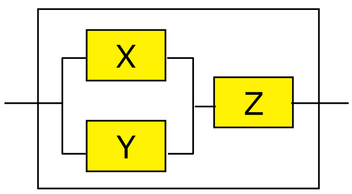
Parallel systems: $A = 1 - \prod_{k=1..n} \{1 - A_k\}$

Fault trees

- Deductive top-down approach: effects ➢ causes
  - Starting from an undesirable event, represent graphically its possible causes (combinations of events)
  - Combination of events: Logical gates

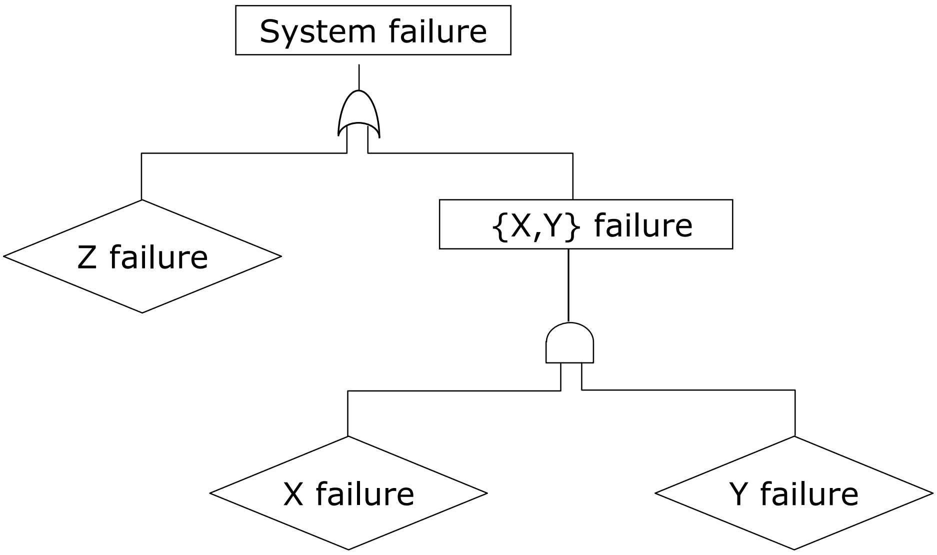


- Example:
  - System: 3 components X, Y, Z
    - X, Y: parallel
    - Z: series with {X,Y}



Elementary events: failures of X, Y, Z
System event: system failure

Example



Model processing

Stochastically independent components

- AND gate
  - Output event E occurs when input events E_1 AND E_2 AND … E_n occur

    $E = E_1 \cap E_2 \cap \dots \cap E_n$

    $Prob.(E) = Prob.(E_1) \,.\, Prob.(E_2) \,.\, \dots \,.\, Prob.(E_n)$

- OR gate
  - Output event E occurs when input event E_1 OR E_2 ... OR E_n occur

    $E = E_1 \cup E_2 \cup \dots \cup E_n$

    $\overline{E} = \overline{E_1} \cap \overline{E_2} \cap \dots \cap \overline{E_n}$

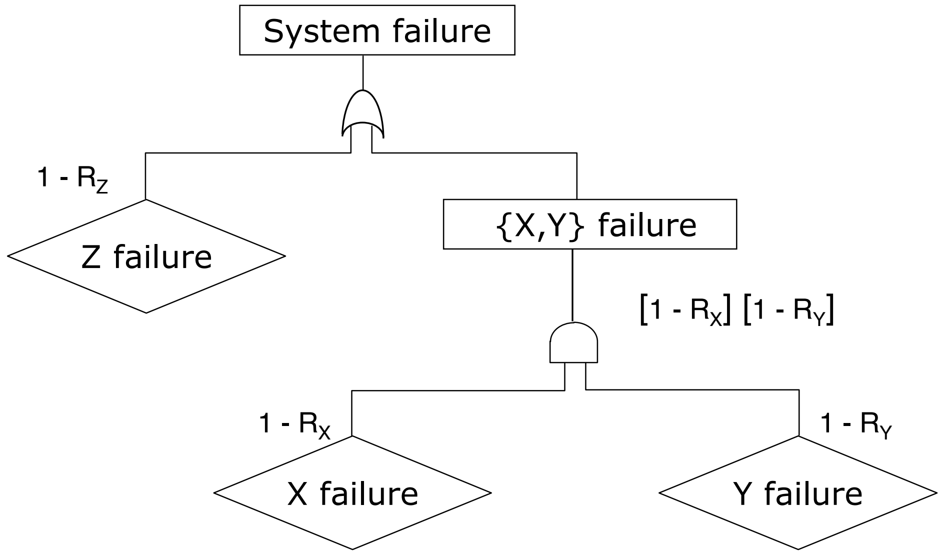
    $Prob.(E) = 1 - [1 - Prob.(E_1)] \,.\, [1 - Prob.(E_2)] \,.\, \dots \,.\, [1 - Prob.(E_n)]$

  - Two elementary events:

    $E = E_1 \cup E_2$

    $Prob.(E) = Prob.(E_1) + Prob.(E_2) - Prob.(E_1).\ Prob.\ (E_2)]$

Example



$$1 - R = 1 - R_Z + [1 - R_X] \,.\, [1 - R_Y] - [1 - R_Z]\,[1 - R_X][1 - R_Y]$$

$$\Downarrow$$

$$R = R_Z\,[R_X + R_Y - R_X\,R_Y]$$

Minimal cut sets

- Cut set
  - set of events whose simultaneous occurrence leads to the occurrence of the top event of the tree
- Minimal cut-set
  - Cut-set that does not include any other
  - Order: number of events of the cut set
    - Order 1: a single event could lead to Top event
- Each minimal cut set of a fault tree describes significant combination of faults that could lead to system failure
  - Critical components
  - Identify design weaknesses ➠ redundancy needs

Minimal cut set computation: Boolean algebra

$A \cap A = A$

$A \cup A = A$

$A \cup B = A$ si $A \supset B$

$A \cap B = B$ si $A \supset B$

$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

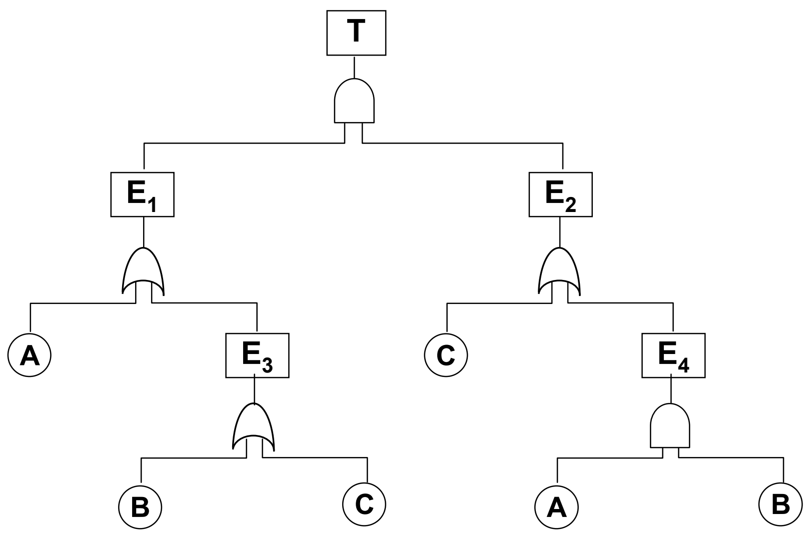
$\overline{A \cup B} = \overline{A} \cap \overline{B}$

$\overline{A \cap B} = \overline{A} \cup \overline{B}$

$A \cup (\overline{A} \cap B) = A \cup B$

$A \cap (\overline{A} \cup B) = A \cap B$

Minimal cut sets: example



$E_3 = B \cup C$

$E_1 = A \cup (B \cup C) = A \cup B \cup C$

$E_4 = A \cap B$

$E_2 = C \cup (A \cap B)$

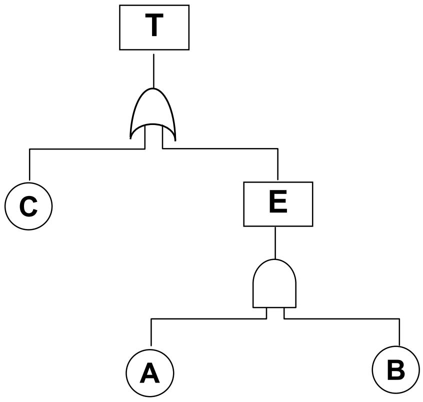
$T = E_1 \cap E_2 = (A \cup B \cup C) \cap (C \cup (A \cap B))$

$T = (A \cup B \cup C) \cap C \cup (A \cup B \cup C) \cap (A \cap B)$

$T = (A \cap C) \cup (B \cap C) \cup C \cup (A \cap B) \cup (A \cap B) \cup (A \cap B \cap C)$

$T = C \cup (A \cap B)$

Reduced fault tree



Prob.{T} = Prob.{C∪(A∩B)}

= Prob.{C}+Prob.{A} Prob.{B} - Prob.{A} Prob.{B} Prob.{C}

Cut sets: Reliability computation

C_i minimal cut set - ordre m_i : $C_i = E1_i \cap E2_i \cap \dots \cap Em_i$

Em_i : basic events T : top event

$$\text{Prob.}\{T\} = P\{C_1 \cup C_2 \cup \dots C_m\}$$

$$\text{Prob.}\{T\} = \sum_{i=1}^{m} \text{Prob.}\{C_i\} - \sum_{j=2}^{m} \sum_{i=1}^{j-1} \text{Prob.}\{C_i \cap C_j\} + \sum_{k=3}^{m} \sum_{j=2}^{k-1} \sum_{i=1}^{j-1} \text{Prob.}\{C_i \cap C_j\} + \dots (-1)^m \, \text{Prob.}\{C_i \cap C_j \dots \cap C_m\}$$

If probability of occurrence of basic events small:

$$\text{Prob.}\{T\} \approx \sum_{i=1}^{m} \text{Prob.}\{C_i\}$$

Prob.(T) bounds:

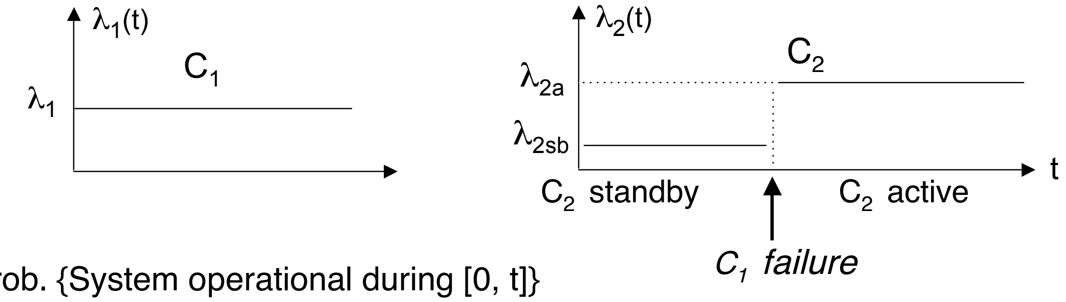
$$\boxed{\sum_{i=1}^{m} \text{Prob.}\{C_i\} - \sum_{j=2}^{m} \sum_{i=1}^{j-1} \text{Prob.}\{C_i \cap C_j\} \le \text{Prob.}\{T\} \le \sum_{i=1}^{m} \text{Prob.}\{C_i\}}$$

Reliability block diagrams & Fault trees

- Pros
  - useful support to understand system failures and relationships with components failures
  - Model processing is easy: powerful tools exist
  - Helpful to identify weak points in the design
- Cons
  - Components and events should be stochastically independent
    - some extensions take into account some kinds of dependencies (e.g., extended fault trees)

Stochastic dependencies: example

- Example:
  - system with two components: C1 (primary), C2 (standby)



R(t) = Prob. {System operational during [0, t]}

- C_1 operational during [0, t] $\Rightarrow$ $\exp(-\lambda_1 t)$
- C_1 fails at τ <u>AND</u> C_2 nonfailed during $[0, \tau]$ <u>AND</u> C_2 operational during $[\tau, t]$

$$\Rightarrow \int_0^t \lambda_1 \exp(-\lambda_1 \tau)\, d\tau \exp(-\lambda_{2sb}\tau) \exp(-\lambda_{2sb}(t-\tau))$$

$$R(t) = \exp(-\lambda_1 t) + \lambda_1 \exp(-\lambda_{2a} t) \int_0^t \exp\left(-(\lambda_1 + \lambda_{2sb} - \lambda_{2a})\tau\right) d\tau$$

$$\boxed{R(t) = \exp(-\lambda_1 t) + \lambda_1 \exp(-\lambda_{2a} t) \left\{ (1 - \exp\left(-(\lambda_1 + \lambda_{2sb} - \lambda_{2a})t\right\} / (\lambda_1 + \lambda_{2sb} - \lambda_{2a}) \right.}$$

State-based models

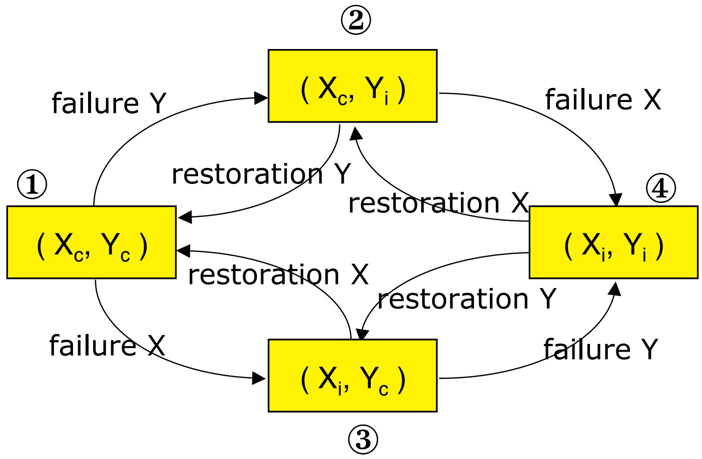
- Example:
  - system: two components X,Y; 1 repairman per component
  - Component states:
    - X_c, Y_c (correct service); X_i, Y_i (incorrect service)
  - System states: (X_c, Y_c), (X_i, Y_c), (X_c, Y_i), (X_i, Y_i)

Nonredundant system:

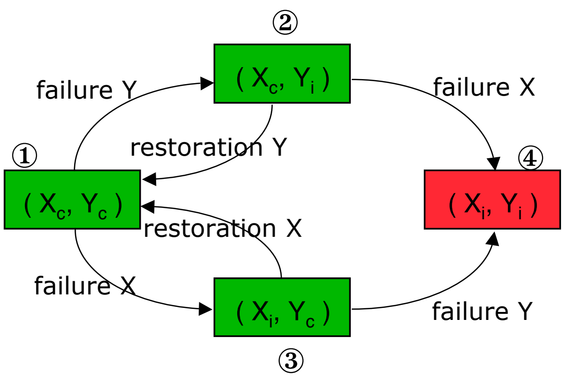
- Correct service: ①
- Incorrect service: ②, ③, ④

Redundant system:

- Correct service: ①, ②, ③
- Incorrect service: ④

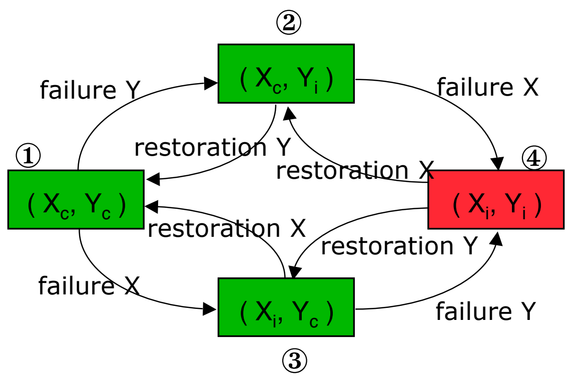


Reliability model



$$R(t) = P_1(t) + P_2(t) + P_3(t)$$

Availability model



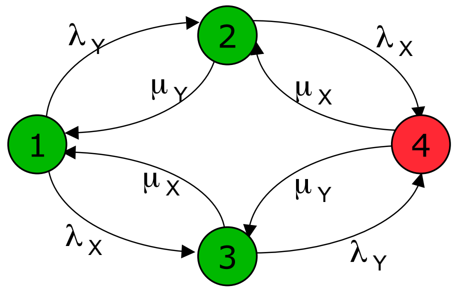
$$A(t) = P_1(t) + P_2(t) + P_3(t)$$

$P_k(t)$: probability system in state k at t

- Computation of Pj(t) depends on the probability distributions associated to state transitions
- Homogeneous Markov chains: constant transition rates

Homogeneous Markov chains

failure rates: λ_X, λ_Y
restoration rates: μ_X, μ_Y



Only one transition might occur during [t, t+dt]:

$P_1(t+dt) = [1 - (\lambda_X\, dt + \lambda_Y\, dt)]\, P1(t) + \mu_Y\, dt\, P_2(t) + \mu_X\, dt\, P_3(t)$

- $1 - (\lambda_X\, dt + \lambda_Y\, dt)$ = Prob.{stay in ① during [t, t+dt]}
- $\mu_Y\, dt$ = Prob.{ transition ② to ① during [t, t+dt]}
- $\mu_X\, dt$ = Prob.{ transition ③ to ① during [t, t+dt]}

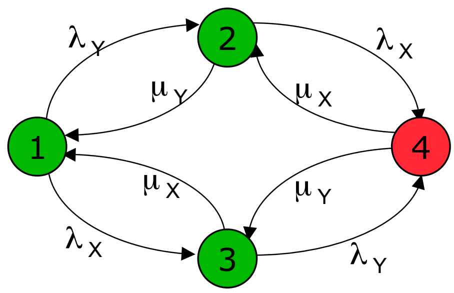
$P_2(t+dt) = \lambda_Y\, dt\, P_1(t) + [1 - (\lambda_X\, dt + \mu_Y\, dt)]\, P_2(t) + \mu_X\, dt\, P_4(t)$

$P_3(t + dt) = \lambda_X\, dt\, P_1(t) + [1 - (\lambda_Y\, dt + \mu_X\, dt)]\, P_3(t) + \mu_Y\, dt\, P_4(t)$

$P_4(t + dt) = \lambda_X\, dt\, P_2(t) + \lambda_Y\, dt\, P_3(t) + [1 - (\mu_X\, dt + \mu_Y\, dt)]\, P_4(t)$

Availability computation

$$\begin{cases} P'_1(t) = -(\lambda_X + \lambda_Y)P_1(t) + \mu_Y P_2(t) + \mu_X P_3(t) \\ P'_2(t) = \lambda_Y P_1(t) - (\lambda_X + \mu_Y)P_2(t) + \mu_X P_4(t) \\ P'_3(t) = \lambda_X P_1(t) - (\lambda_Y + \mu_X)P_3(t) + \mu_Y P_4(t) \\ P'_4(t) = \lambda_X P_2(t) + \lambda_Y P_3(t) - (\mu_X + \mu_Y)P_4(t) \end{cases}$$



Matrix form:

$$\boldsymbol{P}'(t) = \boldsymbol{P}(t)\,.\,\Lambda$$

$$\Lambda = \begin{pmatrix} -(\lambda_X + \lambda_Y) & \lambda_Y & \lambda_X & 0 \\ \mu_Y & -(\lambda_X + \mu_Y) & 0 & \lambda_X \\ \mu_X & 0 & -(\lambda_Y + \mu_X) & \lambda_Y \\ 0 & \mu_X & \mu_Y & -(\mu_X + \mu_Y) \end{pmatrix}$$

$\boldsymbol{P}(t)$: state probability vector
Λ : Infinitesimal generator matrix
(transition rate matrix)

Solution: $\boldsymbol{P}(t) = \boldsymbol{P}(0)\,.\,\exp(-\Lambda t)$

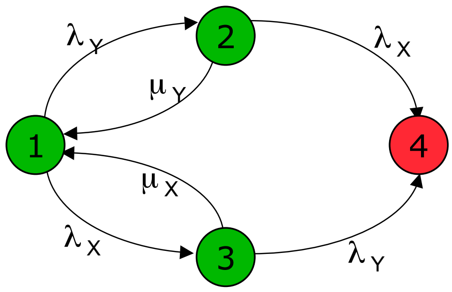
$$A(t) = P_1(t) + P_2(t) + P_3(t) = \boldsymbol{P}(t)\,.\,1_3^A$$

$1_3^A = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$: summation vector

Reliability computation

$$\begin{bmatrix} P'_1(t) & P'_2(t) & P'_3(t) \end{bmatrix} = \begin{bmatrix} P_1(t) & P_2(t) & P_3(t) \end{bmatrix} \begin{bmatrix} -(\lambda_X+\lambda_Y) & \lambda_Y & \lambda_X \\ \mu_Y & -(\lambda_X+\mu_Y) & 0 \\ \mu_X & 0 & -(\lambda_Y+\mu_X) \end{bmatrix}$$

$$\boxed{P'_c(t) = P_c(t) \,.\, \Lambda_{cc}}$$



$P_c(t)$: correct service states probability vector

Λ_{cc} : transition rate matrix — correct service states

Solution: $\boxed{P_c(t) = P(0) \,.\, \exp(-\Lambda_{cc} t)}$

$$\boxed{R(t) = P_1(t) + P_2(t) + P_3(t) = P_c(t) \,.\, 1_3^R}$$

$1_3^R = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$: summation vector

Generalization: m states

Transition rate matrix: $\Lambda = [\lambda_{jk}]$

- λ_{jk} $j \neq k$: transition rate between states j and k (off-diagonal terms)
- $\lambda_{jj}, = -\sum_{k=1,\, k\neq j}^{m} \lambda_{jk}$ $j \neq k$: diagonal terms

$$\Lambda = \begin{array}{c|c} \Lambda_{cc} & \Lambda_{ci} \\ \hline \Lambda_{ic} & \Lambda_{ii} \end{array}$$

	Correct service	Incorrect service		
Correct service	Λ_{cc}	Λ_{ci}	m_c states	$m_c + m_i = m$
Incorrect service	Λ_{ic}	Λ_{ii}	m_i states	

State probability vector:

$P(t) = (P_1(t) \;\; P_2(t) \ldots P_m(t))$

$P(t) = (P_c(t) \; P_i(t))$

$P_c(t) = (P_1(t) \;\; P_2(t) \ldots P_{mc}(t))$ $\qquad$ $P_i(t) = (P_{mc+1}(t) \;\; P_{mc+2}(t) \ldots P_m(t))$

Quantitative measures: summary

- $A(t) = P(0) \cdot exp(\Lambda t) \cdot 1^A_{mc}$

  $1^A_{mc} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ mc "1", mi "0"

- $A = \lim_{t=\infty} A(t) = \Pi_c \cdot 1^A_{mc}$
  - $\Pi_c = (0\ \ 0\ \ \dots\ \ 0\ \ 1_j\ 0\ \ \dots\ \ 0) \cdot \Lambda_j^{-1} \cdot 1^A_{mc}$
  - Λ_j = obtained from Λ by replacing jth column by "1"

- $R(t) = P_c(0) \cdot exp(\Lambda_{cc} t) \cdot 1_{mc}$

  $1_{mc} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ mc "1"

- $MTTF = -P_c(0) \cdot \Lambda_{cc}^{-1} \cdot 1_{mc}$

  $1_{mi} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ mi "1"

- $MTTR = -P_i(0) \cdot \Lambda_{ii}^{-1} \cdot 1_{mi}$

- $MUT = \dfrac{\Pi_c \cdot 1_{mc}}{\Pi_c \cdot \Lambda_{ci} \cdot 1_{mi}}$

  $MUT + MDT = MTBF = \dfrac{1}{\Pi_c \cdot \Lambda_{ci} \cdot 1_{mi}}$

- $MDT = \dfrac{\Pi_i \cdot 1_{mi}}{\Pi_c \cdot \Lambda_{ci} \cdot 1_{mi}}$

Markov reward models

- Useful for combined performance-availability evaluation ("performability")
- Extension of continuous time Markov chains with rewards
  - Reward: performance index, capacity, cost, etc.
- Quantitative measures
  - r_i = reward rate associated with *state i* of the Markov chain
  - $Z(t) = r_{X(t)}$: instantaneous reward rate of Markov chain X(t)

    Expected instantaneous reward rate: $E[Z(t)] = \sum r_i \cdot P_i(t)$

    Expected steady-state reward rate: $\lim_{t=\infty} E[Z(t)] = \sum r_i \cdot \pi_i$

  - $Y(t)$ = accumulated reward in [0, t]

    $Y(t) = \int_0^t Z(x) \cdot dx$ $\qquad$ $E[Y(t)] = \sum r_i \cdot \int_0^t P_i(x) \cdot dx$

Modelling of complex systems

- Model largeness
  - Number of components, dependencies
- Stiffness
  - parameters on different time scales

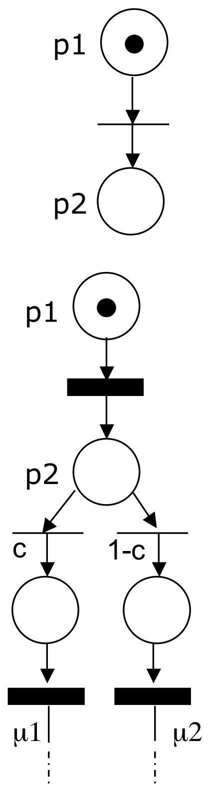
- Largeness/stiffness tolerance and avoidance techniques
  - automatic generation of state space using Stochastic Petri nets and their extensions
  - Structured model composition approaches with explicit description of dependencies
  - Hierarchical model decomposition and aggregation
  - Robust model solution and efficient storage techniques

- Non exponential distributions

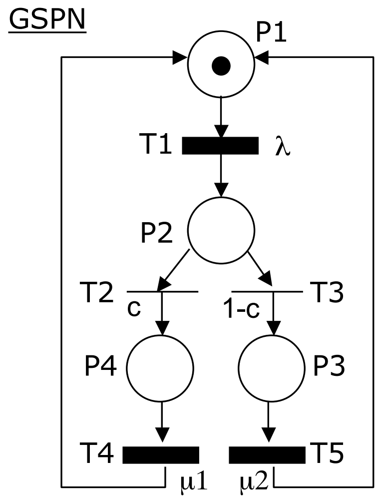
  - Use of semi-Markov or non Markovian models
  - Approximation by exponential models using method of stage
  - Use of simulation

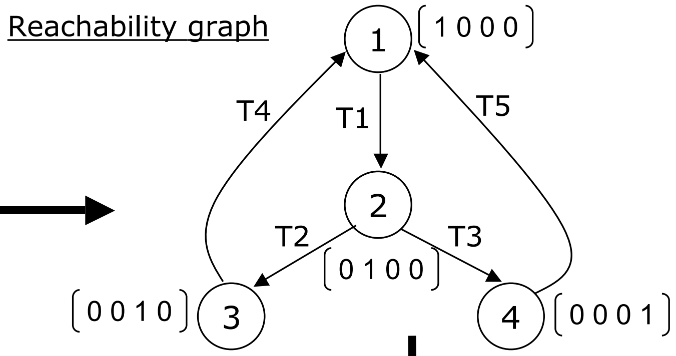
Petri nets, SPNs and GSPNs

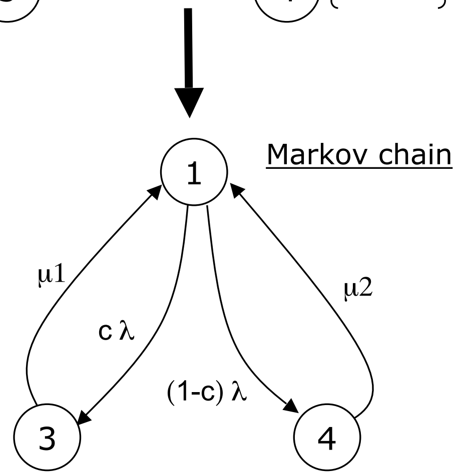
- A Petri net (PN) is defined by (P, T, I, O, M)
  - P: Places, represent conditions in the system
  - T: Transitions, represent events
  - I, O: Input, Output arcs connecting places and transitions
  - M: initial marking, number of tokens in each place
- Stochastic Petri nets (SPNs)
  - PNs with exponentially distributed timed transitions
- GSPNs (Generalized Stochastic Petri Nets)
  - PNs with exponentially distributed timed transitions and instantaneous transitions
- Advantages
  - Suitable for describing concurrency, synchronization, …
  - Reachability graph isomorphic to a Markov chain



Example 1: computer system with two failure and restoration modes (imperfect detection)



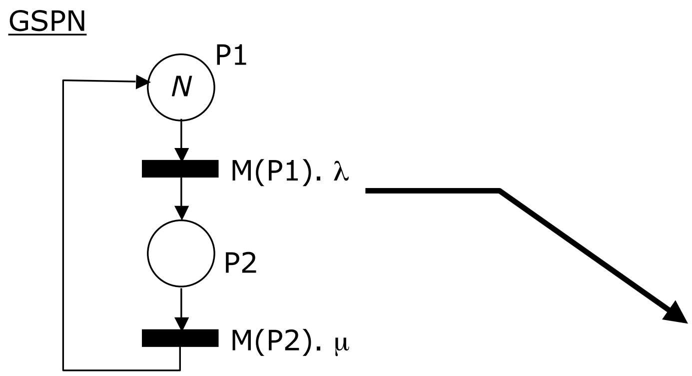


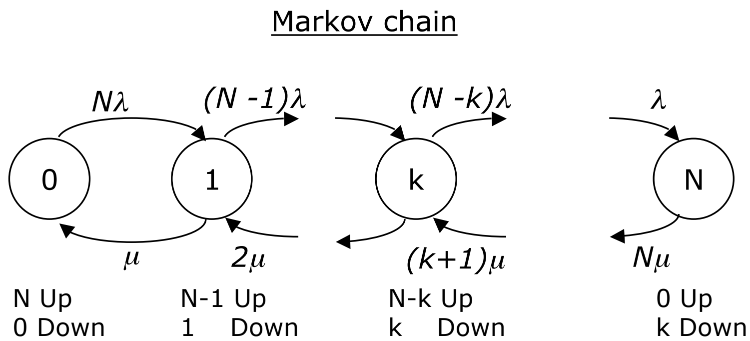


P1: system OK
P2: error activated
P3: nondetected error
P4: detected error

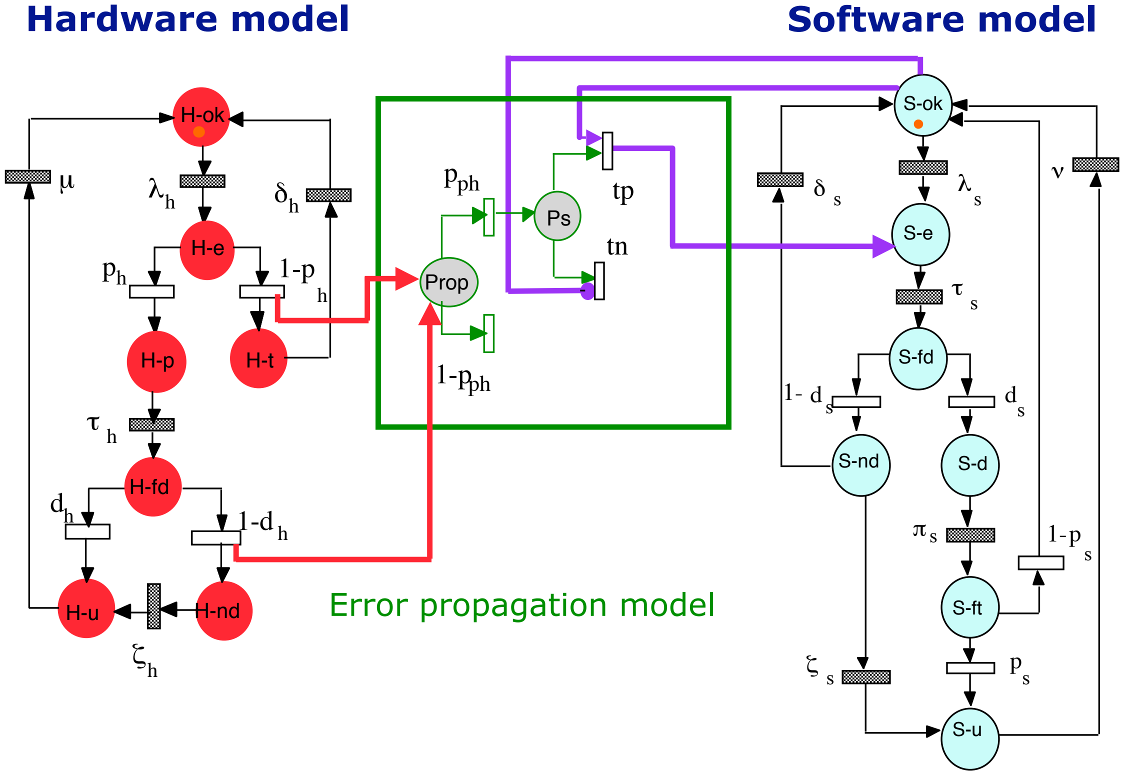
T1: fault activation
T2: detection
T3: nondetection
T4: restoration
T5: periodic maintenance

Example 2: *N* redundant component system, 1 repairman per component





Hardware- software with error propagation



Block modeling approach

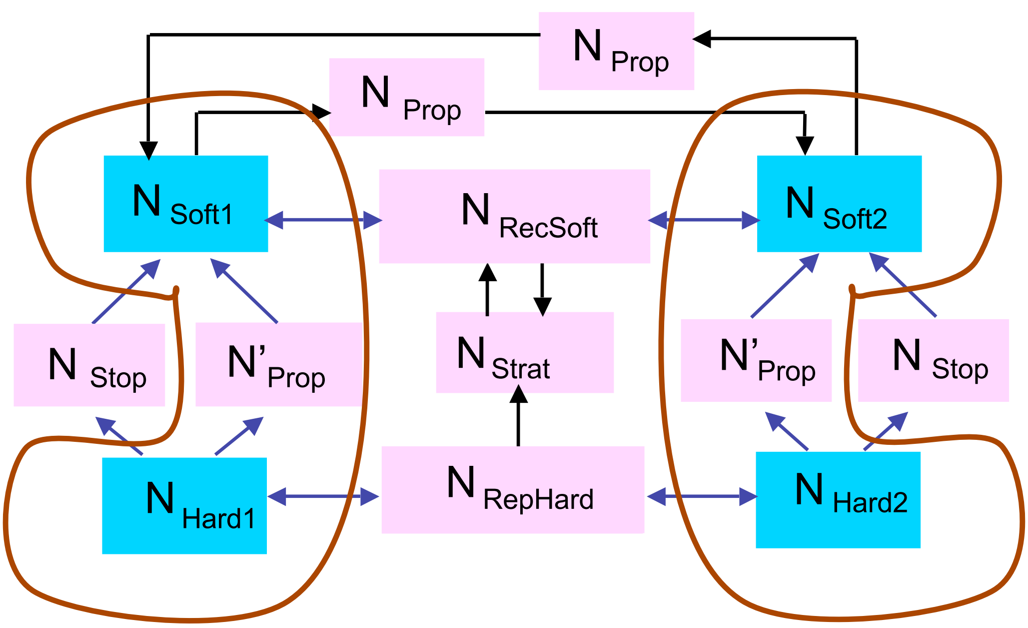
- Structured composition modeling of complex systems with explicit description of dependencies
  - Dependencies:
    - functional, structural, due to maintenance or fault tolerance strategies
- Block-Model (high-level model)
  - Blocks ⇒ model
    - Components behavior
    - Dependency between components
  - Arrows: interactions
- Detailed model
  - Block ⇒ GSPN
- Application to CAUTRA:French air traffic control comp. Syst.
  - Comparative availability analysis of 16 alternative architectures

Illustration: Duplex System

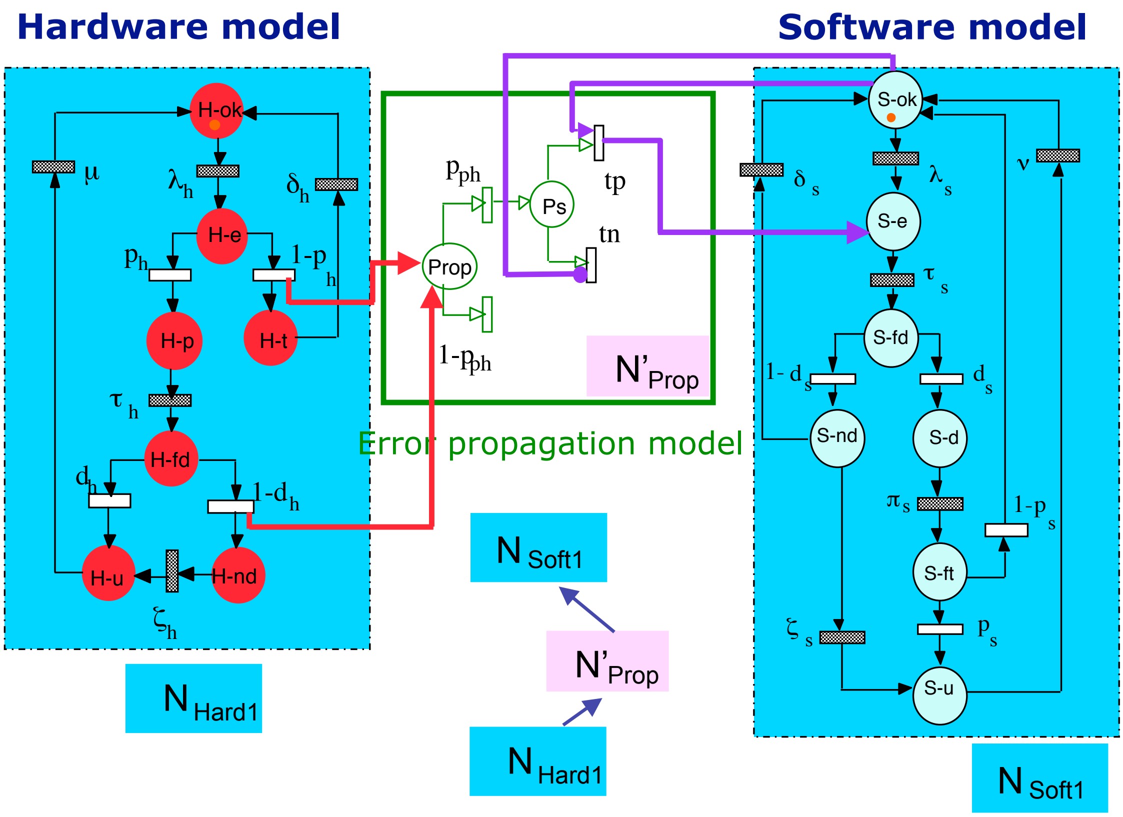
- Composition
  - Two software replica (principal & spare), with communications
  - Two hardware components
- Software reconfiguration after principal replica failure
  - Role switching
- Hardware maintenance and fault tolerance
  - One permanent fault
  - One repairman

- Software ←→ Software
  - Error propagation (Prop)
  - Reconfiguration (RecSoft)
- Hardware ←→ Hardware
  - Sharing of one repairman (RepHard
- Hardware → Software
  - Error propagation (Prop)
  - Stop and restart (Stop)
  - Global reconfiguration strategy (Strat)

Block model of the Duplex system



Block model → GSPN



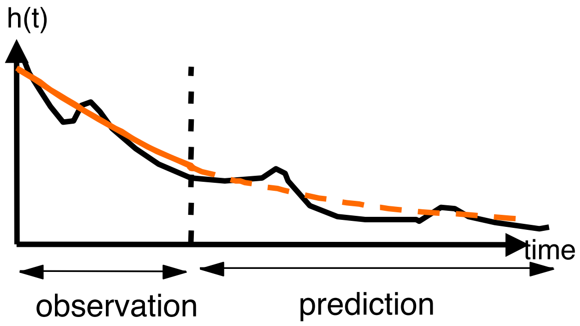
Tools

Surf-2	GSPNs, Markov	LAAS, France
Great-SPN	GSPNs and stochastic well formed nets	Torino, Italy
UltraSAN	Stochastic Activity Networks (SANs)	UIUC, USA
Möbius	Multi-formalism (SANs, PEPA, Fault tree,…	UIUC, USA
SHARPE	Multi-formalism (Combinatorial , state-based) hierarchical models	Duke, USA
DRAWNET++	Multi-formalism (Parametric Fault trees, SWN)	U. del Piemonte orientale, U.Torino, U. Napoli, Italy
SPNP	Multi-formalism (SPNs, Stochastic Reward nets, NonMarkovian, fluid models)	Duke, USA
DEEM	Deterministic and SPNs, Multi-phased systems	UNIFI-PISA, Italy
TimeNET	nonMarkovian SPNs	Hamburg, Germany
DSPNexpress	Deterministic and stochastic Petri nets	Dortmund, Germany

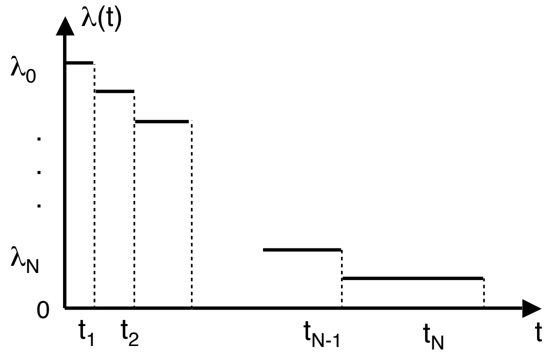
ADVISER, ARIES, CARE III, METFAC, SAVE, SURE, ASSIST, HARP, etc..

Software Reliability Growth Models

- N(t) = Number of failures during [0, t]
- Failure intensity: $h(t) = \dfrac{d}{dt} E[N(t)]$

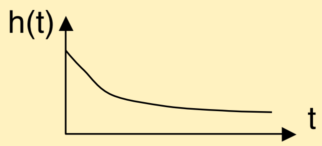
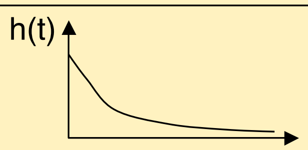
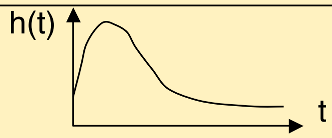
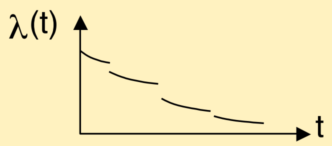


NHPP models:
N(t) described as a Non
Homogeneous Poisson Process

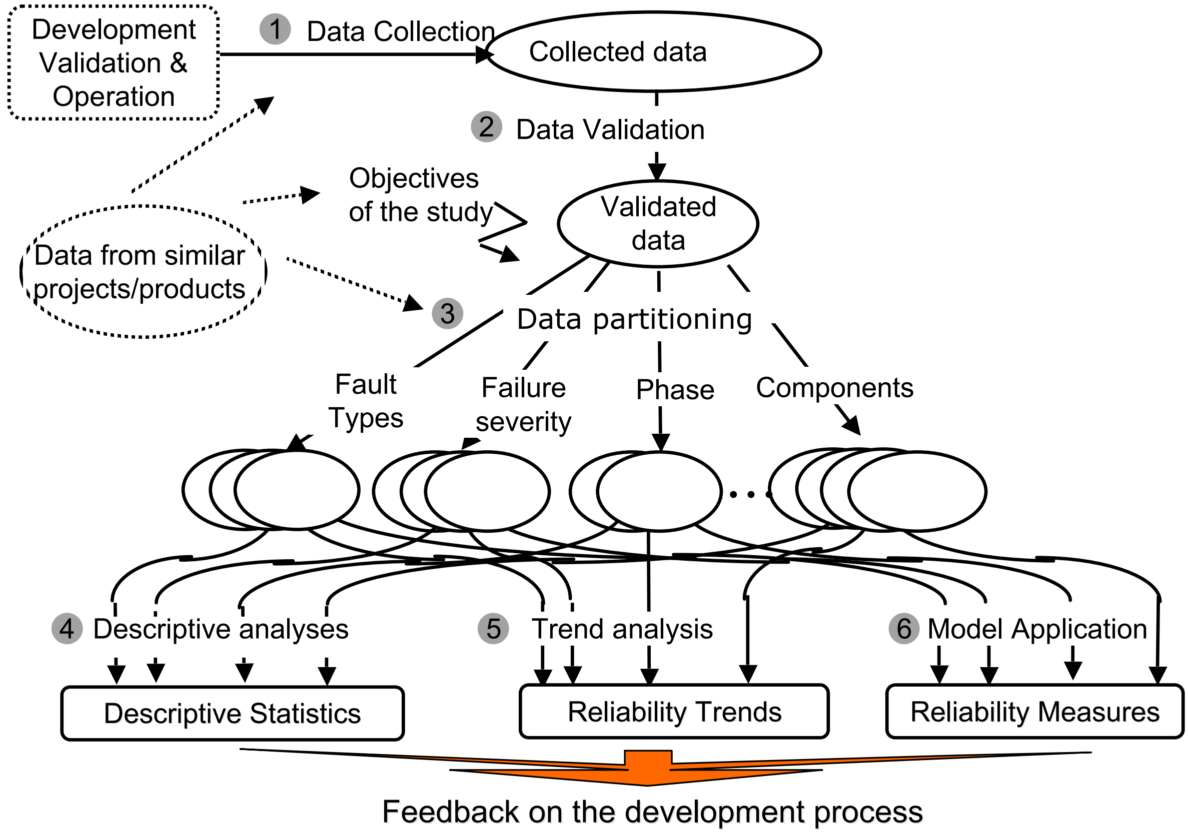


Models with decreasing
failure rates

Reliability growth models: Examples

Model	**h(t) or λ(t) shape**
Hyperexponential $h(t) = \dfrac{\omega\zeta_{sup}\, e^{-\zeta_{sup} t} + \varpi\zeta_{inf}\, e^{-\zeta_{inf} t}}{\omega e^{-\zeta_{sup} t} + \varpi e^{-\zeta_{inf} t}}$	
Exponential $h(t) = N\,\phi \exp(-\phi\, t)$	
S-Shaped $h(t) = N\,\phi^2\, t \exp(-\phi\, t)$	
Doubly Stochastic $\lambda_i(t) = \dfrac{\alpha}{t + \psi(i)}$ $\quad \psi(i) = \beta_1 + \beta_2 i$	

Global method



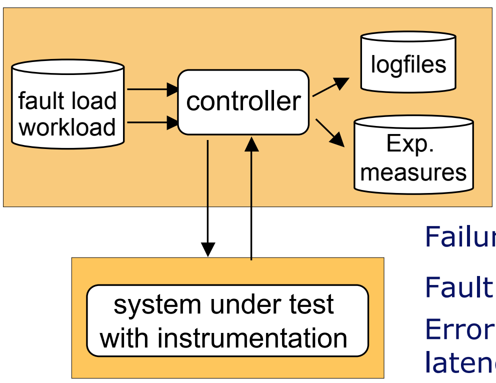
Models calibration and validation - data

- Estimation of model parameters based on data collected from operation, controlled experiments or using expert judgments

data processing and statistical analysis techniques

Controlled experiments

(testing, fault injection)

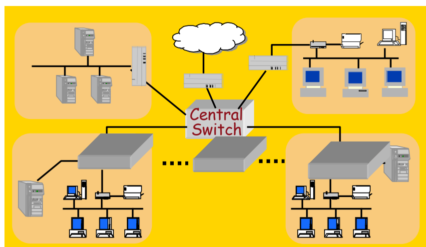


Failure modes

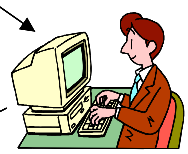
Fault coverage

Error detection latency

data collection in operation

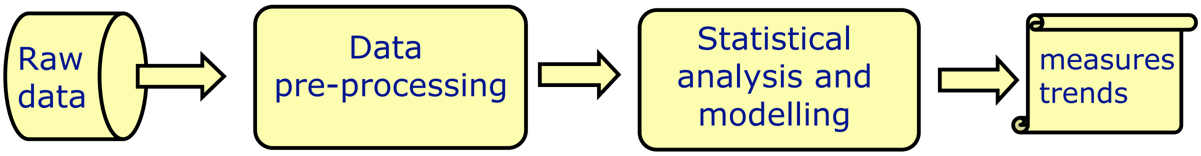


event logs
failure reports
(automatic/manual)



Failure rates,
Repair rates, Failure modes

Assessment based on operational data

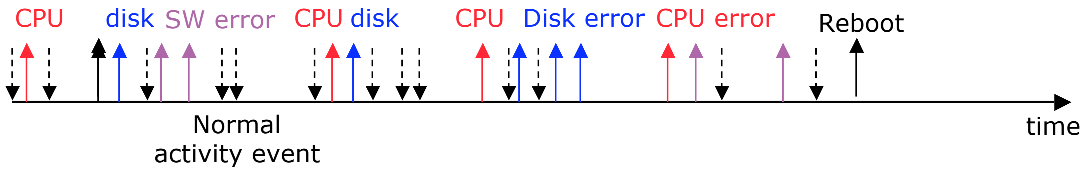


- event logs
- failure reports

- Extraction of relevant information
- identification/categorization of errors, failures, reboots
- occurrence times, durations

(data clustering algorithms)

- Failure/error distributions
- Failure/error recovery rates Availability estimation
- Impact of workload
- Error propagation analysis



- Measurement-based studies and trends
  - Trends: Hardware, Software, distributed systems and middleware, Internet, Human-computer interaction, security, wireless, etc..
  - Systems: FTMP, SIFT, TANDEM, VAX, SUNOS/Solaris, Windows NT/2K, Linux, Symbian OS..

Examples (1)

- DEC VAXCluster Multicomputer
  - 7 processing nodes et 4 disk controllers connected through a bus
  - 8 months (december 1987 — August 1988)

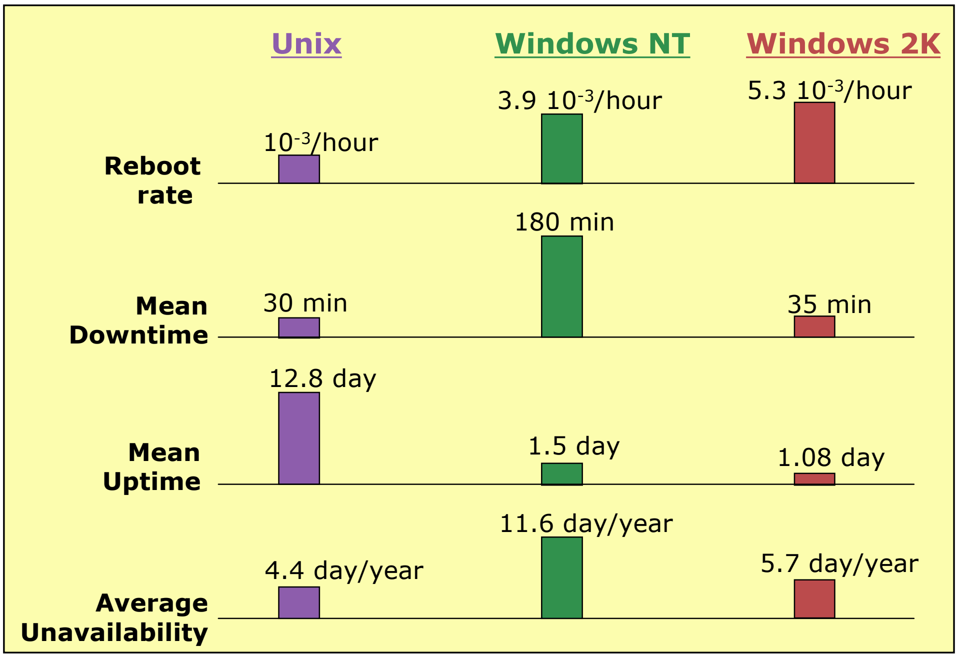
	MTTF	λ	MTTR	μ	coverage
CPU	8400 h	$1.19\ 10^{-4}$ /h	24.8 min	2.42 /h	0.970
Disk	656 h	$1.52\ 10^{-3}$ /h	110 min	0.54 /h	0.997
Network	1400 h	$7.14\ 10^{-4}$ /h	53.4	1.12 /h	0.991
Software	677 h	$1.48\ 10^{-3}$ /h	24.4	2.46 /h	0.1

- CMU Andrew file server
  - 13 SUN II workstations - collection period: 21 workstation.year

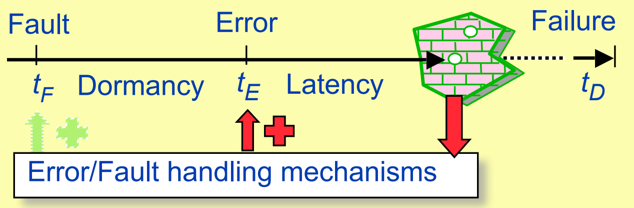
	Mean time to occurrence (per system)	Number of events (all systems)
Permanent failures	6552 h	29
Intermittent faults	58 h	610
Transient faults	354 h	446
System crashes	689 h	298

Assessment based on operational data (2)

- LAAS-CNRS local area network
  - 418 SunOS/Solaris, 78 Windows NT, 130 Windows 2K
  - Jan. 1999-Oct. 2003: 1392 system.year - 50 000 reboots



Fault tolerance efficiency assessment



Fault tolerance coverage
C = Prob.{correct service delivery/fault}

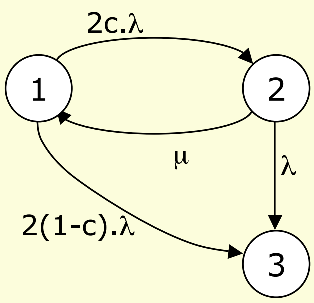
Error and Fault handling Coverage

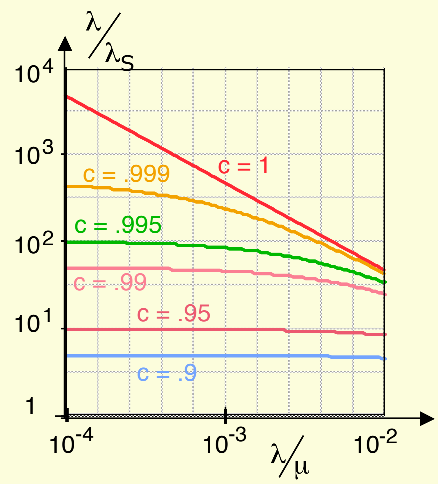
Assumption Coverage

fault tolerance coverage impact

Duplex system (hot standby)

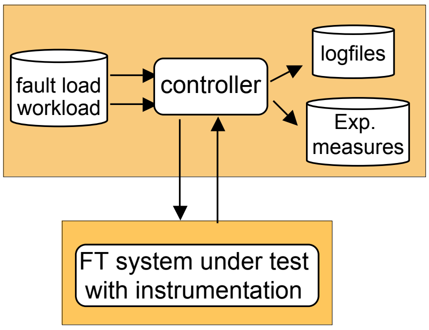
Component failure rate: λ
Component repair rate: μ
system failure rate: λ_S



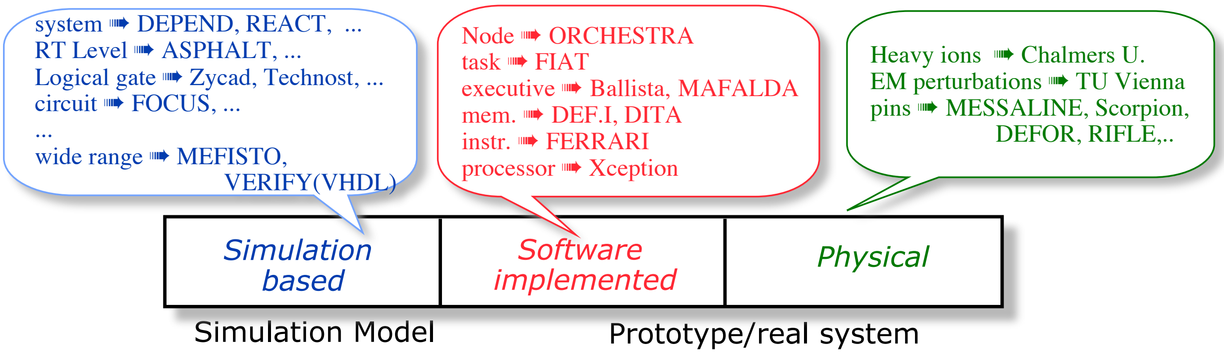


Experimental assessment

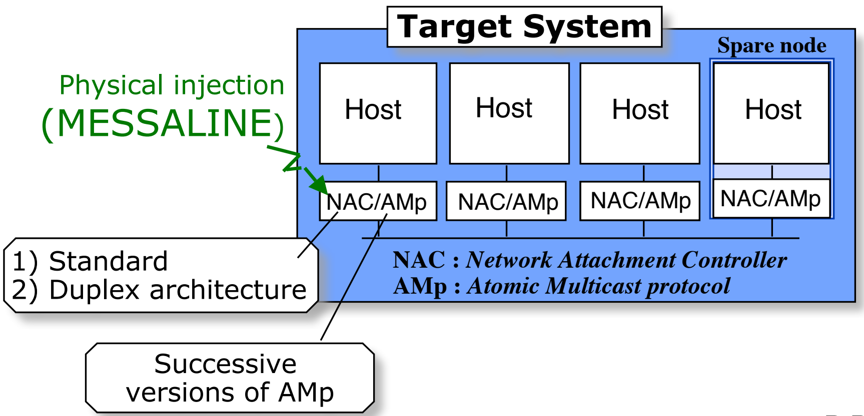
- Fault injection target
  - HW, drivers, OS, API, middleware, application
- Fault model
  - Bit-flips (data, code segments, parameters)
  - instruction mutation, dropping messages, …



- Fault injection techniques

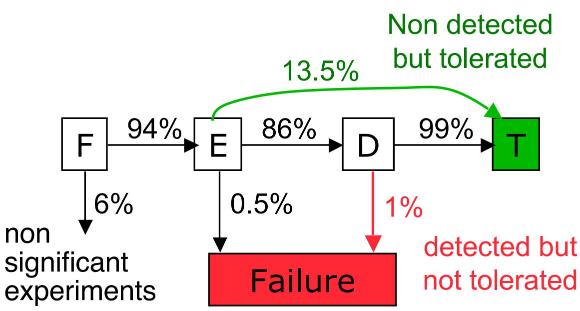


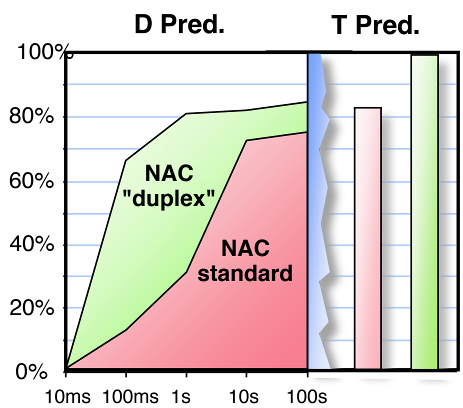
Delta-4 project



T Predicate:
Protocol Properties OK and errors confined

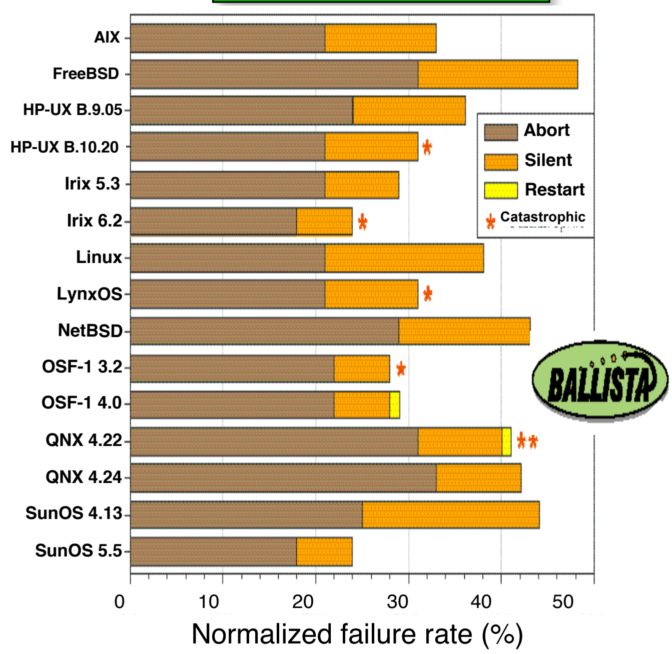
D Predicate:
Auto-extraction of faulty node





POSIX OS robustness testing (BALLISTA)

15 « (C)OTS » OSs
[Koopman & DeVale 99 (FTCS-29)]



POSIX System call parameter mutation
(233 functions)

Dependability benchmarking

- "Standardised" framework for evaluating dependability and performance related measures experimentally or based on experimentation and modeling
  - Characterize objectively system behavior in presence of faults
  - Non-ambiguous comparison of alternative solutions
- Non-ambiguity, confidence, acceptability ensured by a set of properties:
  - Representativeness, Reproducibility, Repeatability, Portability, Non-intrusiveness, Scalability, Cost effectiveness
- Benchmark = specification of a set of elements (dimensions) and a set of procedures for running experiments on the benchmark target to obtain dependability measures
- DBench IST project (www.laas.fr/dbench)
- SIGDeb: Special Interest Group on Dependability Benchmarking (IFIP 10.4 WG)

DBench: Benchmarks developed

- General purpose operating systems
  - Robustness and timing measures, TPC-C Client, faulty application
- Real-time kernels in onboard space system
  - Predictability of the kernel response time, faulty application
- Engine control applications in automotive systems
  - Impact of application failures on system safety, transient hardware faults
- On-line transaction processing (OLTP) environments
  - TPC-C-based, operator, software & hardware faults
  - Web-servers, SPEC-based, operator, software & hardware faults

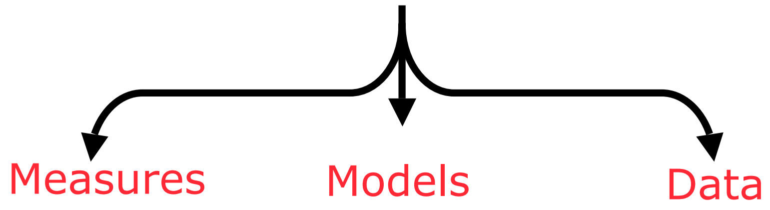
Evaluation wrt. malicious threats

- Historically, attention has been mainly focused on prevention and protection approaches, and less on evaluation
- Traditional evaluation methods
  - Qualitative Evaluation criteria
    - TCSEC (USA), ITSEC (Europe), Common Criteria
    - Security levels based on functional and assurance criteria
  - Risk assessment methods
    - Subjective evaluation of vulnerabilities, threats and consequences
  - Red teams: try to penetrate or compromise the system

  ➟ Not well suited to take into account the dynamic evolution of systems, their environment and threats during operation, and to support objective design decisions
- Need for security quantification approaches similar to those used in dependability relative to accidental faults

Security quantification challenges

- Defining representative measures
  - Are new measures needed?
- Modeling attackers behaviors and system vulnerabilities and assessing their impact on security properties
  - How different is it, compared to modeling accidental faults and their consequences?
- Elaborating representative assumptions
  - Continuous evolution of threats and attackers behaviors
  - Need for unbiased and detailed data



Measures and Models

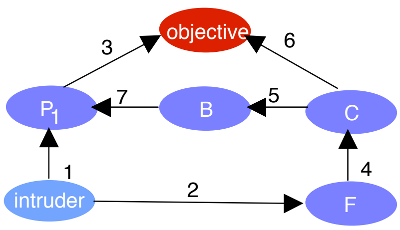
- Feasibility of a probabilistic security quantification explored early in the 1990's (PDCS and DeVa projects)
  - Measure = effort needed for a potential attacker to defeat the security policy [City U.]
  - Preliminary experiments using tiger teams [Chalmers U.]
  - A "white-box" approach for modeling system vulnerabilities and quantifying security, using "privilege graph" [LAAS-CNRS]
- Graph-based models for the description of attack scenarios
  - Attack graphs, attack trees, etc.
- Stochastic state-based models to assess intrusion tolerant syst.
  - DPASA "Designing Protection and Adaptation into a Survivable Arch."
  - SITAR Intrusion Tolerant System [Duke, MCNC]
- Epidemiological malware propagation models
- Complex network theory, game theory, etc.

LAAS quantitative evaluation approach

- Motivation
  - Take into account security/usability trade-offs
  - Monitor security evolutions according to configuration and use changes
  - Identify the best security improvement for the least usability change
- Probabilistic modeling framework
  - Vulnerabilities
  - Attackers
- Measure = effort needed for a potential attacker to defeat the security policy
- Application to Unix-based systems

Overview

Vulnerabilities Modeling privilege graph



Node = set of privileges

Arc = vulnerability class

path = sequence of vulnerabilities that could be exploited by an attacker to defeat a security objective

weight = for each arc, effort to exploit the vulnerability

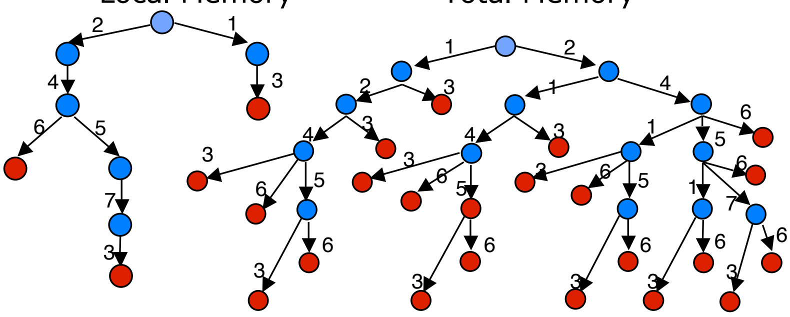
Application to LAAS LAN

ESOPE tool

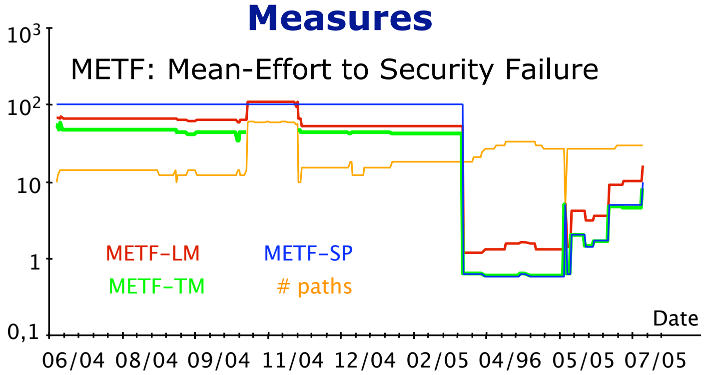
Generation of attack scenarios

Assumption LM
Local Memory

Assumption TM
Total Memory



Measures



Open issues

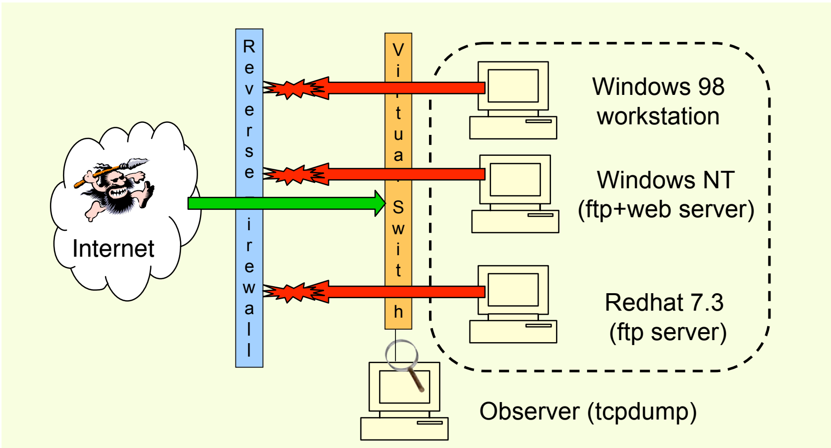
- Is the model valid in the real world?
- TM and ML are two extreme behaviors, but what would be a "real" attacker behavior?
- Weight parameters are assessed arbitrarily (subjective?)
  - Tenacity? Collusion? Attack rates?
- We need real data !!

Data

- Measurement
  - monitoring and collection of real attack data to learn about malicious activities on the Internet
    - Exploited vulnerabilities, attacks tools, propagations, etc.
- Honeypots
  - "an information system resource whose value lies in unauthorized or illicit use of that resource" [Spitzner 2002]
- Internet Telescopes and honeypot based project
  - CAIDA, Internet Motion Sensor, Team Cymru Darknet, …
  - Leurré.com
- Logs sharing
  - Dshield, Internet Storm Center, WorRadar, ……
- Vulnerabilities databases

Leurré.com

- Deploy on the Internet a large number of identically configured low-interaction honeypots at diverse locations
- Carry out analyses based on collected data to better understand threats and build models to characterize attacks



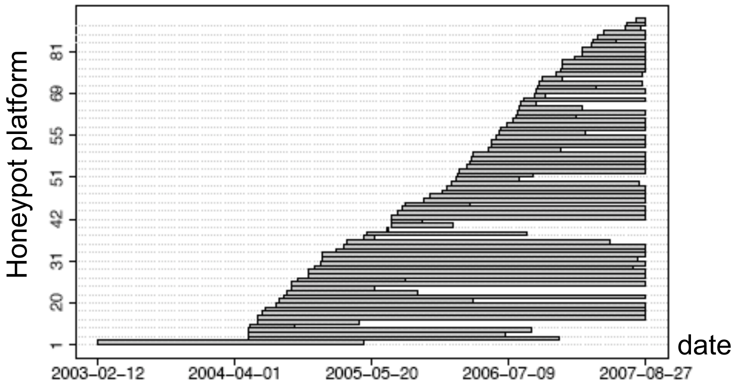
80 honeypots - 30 countries - 5 continents

Win-Win partnership

- The interested partner provides:
  - One old PC (PentiumII, 128M RAM, 233 MHz, …)
  - 4 routable IP addresses
- Eurecom offers
  - Installation CD Rom
  - Remote logs collection and integrity check
  - Access to the whole SQL database
- Collaborative research projects using collected data
  - CADHo: (Eurecom, LAAS, CERT-Renater)
    - Analysis and modeling of attack processes using low interaction honeypots data
    - Development and deployment of high-interaction honeypots to analyze behavior of attackers once they get access and compromise a target

Overview of collected data

- Data collection since 2003
  - 3026962 different IP addresses from more than 100 countries
  - 80 honeypot platform deployed progressively



- Information extracted from the logs
  - Raw packets (entire frames including payloads)
  - IP address of attacking machine
  - Time of the attack and duration
  - Targeted virtual machines and ports
  - Geographic location of attacking machine (*Maxmind, NetGeo*)
  - Os of the attacking machine (*p0f, ettercap, disco*)

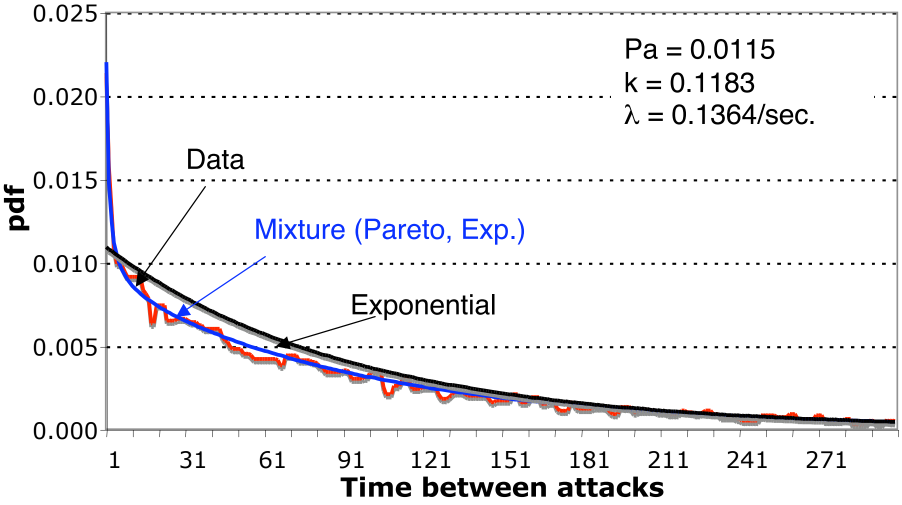
Analysis and modeling of attack processes

- Automatic data analyses to extract useful trends and identify hidden phenomena from the data
  - Categorization of attacks according to their origin, attack tools, target services and machines, etc.
  - Analysis of similarities of attack patterns for different honeypot environments, etc.
  - Publications available at: www.leurrecom.org/paper.htm
- Stochastic modeling of attack processes
  - Identify probability distributions that best characterize attack occurrence and attack propagation processes
  - Analyze whether data collected from different platforms exhibit similar or different malicious attack activities
  - Predict occurrence of new attacks on a given platform based on past observations on this platform and other platforms
  - Publications available at: www.cadho.org

“Times between attacks” distribution

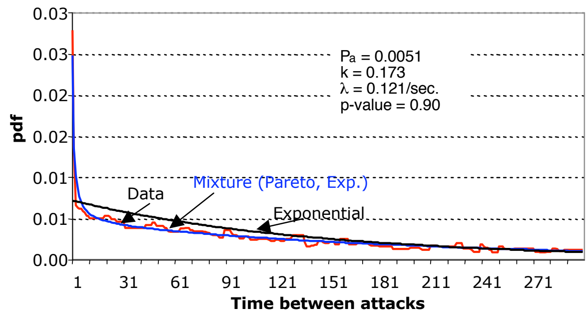
- An attack is associated to an IP address
  - occurrence time associated to the first time a packet is received from the corresponding address
- Best fit provided by a mixture distribution

$$pdf(t) = P_a \frac{k}{(t+1)^{k+1}} + (1 - P_a)\lambda e^{-\lambda t}$$

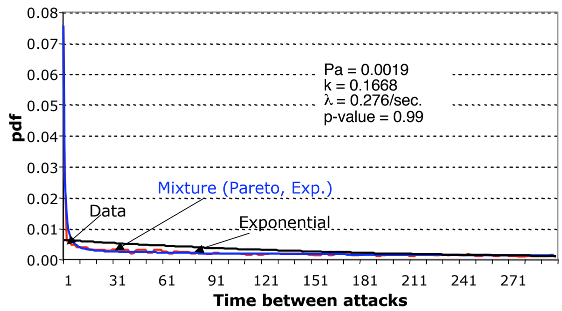


Platform 6

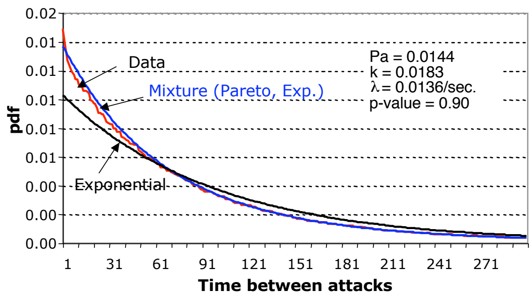
“Times between attacks” distributions (2)



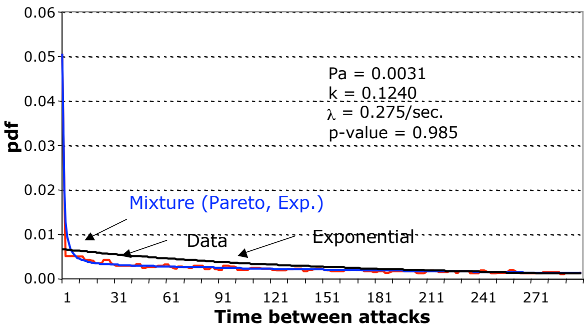
Platform 5



Platform 9



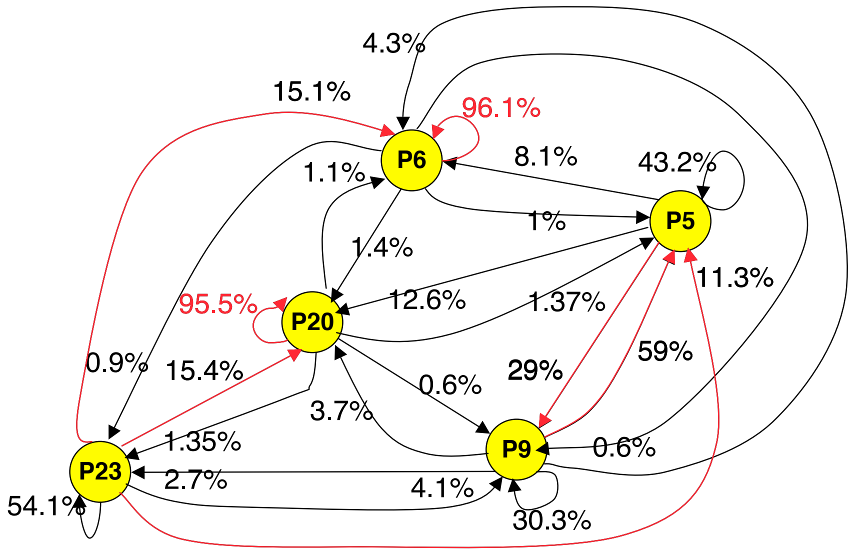
Platform 20



Platform 23

Propagation of attacks

- A Propagation is assumed to occur when an IP address of an attacking machine observed at a given platform is observed at another platform



Discussion

- Preliminary models to characterize attack processes observed on low-interaction honeypots
- Several open issues
  - Need for predictive models that can be used to support decision making during design and operation
  - How to assess the impact of attacks on the security of target systems?
- Honeypots with higher degree of interaction are needed to analyse attacker behavior once they manage to compromise and access to a target
  - first results demonstrate their usefulness and complementarity with low interaction honeypots [Alata et al. 2006]

Resilience evaluation: challenges and gaps

Evaluation wrt accidental faults

Evaluation wrt malicious faults

Unified approaches for resilience evaluation and benchmarking of large systems and critical infrastructures, combining analytical, simulation and experimental techniques

- complexity
- dynamicity
- evolution of threats
- interdependencies
- runtime assessment
- socio, technical and economic dimensions

References

General background

- K. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications* », 2nd Edition, John Wiley and Sons, New York, (2001)
- R.W. Howard, Dynamic Probabilistic Systems, vol. I and vol. II John Wiley & Sons, 1971
- J.-C. Laprie, "Dependability Handbook", CÉPADUES-ÉDITIONS, 1995 (in French)
- B. Haverkort, R. Marie, G. Rubino, K. Trivedi, Performability modeling: Techniques and tools", John Wiley & Sons, ISBN 0-471-49195-0
- M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli and G. Franceschinis, *Modelling with Generalized Stochastic Petri Nets* Wiley Series in Parallel Computing John Wiley and Sons ISBN: 0 471 93059 8 (http://www.di.unito.it/~greatspn/bookdownloadform.html)

Dependability Modeling

- J. Bechta-Dugan, S. J. Bavuso and M. A. Boyd, "Dynamic fault-tree models for fault-tolerant computer systems", *IEEE Transactions on Reliability*, 41, pp.363-377, 1992
- C. Betous-Almeida and K. Kanoun, "Construction and Stepwise Refinement of Dependability Models", *Performance Evaluation*, 56, pp.277-306, 2004
- A. Bondavalli, M. Nelli, L. Simoncini and G. Mongardi, "Hierarchical Modelling of Complex Control Systems: Dependability Analysis of a Railway Interlocking", Journal of Computer Systems Science and Engineering 16(4): 249-261, 2001
- N. Fota, M. Kaâniche, K. Kanoun, "Dependability Evaluation of an Air Traffic Control Computing System," *3rd IEEE International Computer Performance & Dependability Symposium (IPDS-98),* (Durham, NC, USA), pp. 206-215, IEEE Computer Society Press, 1998. Published in./*Performance Evaluation*, Elsvier, 35(3-4), pp.253-73, 1999
- M. Kaâniche, K. Kanoun and M. Rabah, "Multi-level modelling approach for the availability assessment of e-business applications", *Software: Practice and Experience*, 33 (14), pp.1323-1341, 2003
- K. Kanoun, M. Borrel, T. Morteveille and A. Peytavin, "Modeling the Dependability of CAUTRA, a Subset of the French Air Traffic Control System", *IEEE Transactions on Computers*, 48 (5), pp.528-535, 1999

References

Dependability Modeling (cntd)

- I. Mura and A. Bondavalli, “Markov Regenerative Stochastic Petri Nets to Model and Evaluate the Dependability of Phased Missions”, *IEEE Transactions on Computers*, 50 (12), pp.1337-1351, 2001
- M. Rabah and K. Kanoun, “Performability evaluation of multipurpose multiprocessor systems: the "separation of concerns" approach”, *IEEE transactions on Computers*, 52 (2), pp.223-236, 2003
- W. H. Sanders and J. F. Meyer, “Stochastic activity networks: Formal definitions and concepts”, in *Lectures on Formal Methods and Performance Analysis. Lecture Notes in Computer Science* 2090, pp.315-343, Springer-Verlag, 2001
- M. Kaaniche, K. Kanoun, M. Martinello, “User-Perceived Availability of a web-based Travel Agency”, in *IEEE International Conference on Dependable Systems and Networks (DSN-2003), Performance and Dependability Symposium*, (San Francisco, USA), 2003, pp. 709-718.

Software reliability evaluation

- Michael R. Lyu (Ed), *Handbook of Software Reliability Engineering*, Published by IEEE Computer Society Press and McGraw-Hill Book Company, ISBN-10: 0070394008, 1996, (http://www.cse.cuhk.edu.hk/~lyu/book/reliability/)
- J. Musa, A. Iannino, K. Okumoto, *Software Reliability: Measurement, Prediction, Application*. McGraw-Hill, 1987
- B. Littlewood and L. Strigini, "Validation of Ultra-High Dependability for Software-based Systems", Communications of the ACM, vol. 36(11), pp. 69-80, 1993.
- K. Kanoun, J.-C. Laprie, “Software Reliability Trend Analysis: From Theoretical to Practical Considerations,” *IEEE Transactions on Software Engineering*, vol. 9, pp. 740-777, 1994.
- K. Kanoun, M. Kaâniche, J.-C. Laprie, “Qualitative and Quantitative Reliability Assessment,” *IEEE Software*, vol. 14, pp. 77-86, 1997.
- K. Kanoun, M. Kaâniche, C. Béounes, J.C. Laprie, and J. Arlat, “Reliability growth of fault-tolerant software”, IEEE Transactions on Reliability, IEEE Computer Society, 42(2), pp.205-19, 1985.
- J.-C. Laprie, K. Kanoun, “X-ware Reliability and Availability Modeling,” *IEEE Transactions on Software engineering*, vol. SE-18, pp. 130-147, 1992.
- Littlewood, B., P. Popov, L. Strigini, “Assessing the Reliability of Diverse Fault-Tolerant Software-Based Systems”, Safety Science 40: 781-796, 2002

References

Experimental measurements and Benchmarking

- P. Koopman, J. DeVale, “The exception handling effectiveness of POSIX Operating Systems”, IEEE Trans. On Softwrae Engineering, vol. 26, n°9, 2000
- J. Arlat et al., Fault Injection for Dependability Evaluation: A Methodology and some applications”, IEEE Transactions on Software Engineering, vol. 16, n°2, 1990
- J. Carreira, H. Madeira, , J. G. Silva, “Xception: A Technique for the Evaluation of Dependability in Modern Computers”, IEEE Transactions on Software Engineering, vol.24, n°2, 1998
- Eric Marsden, Jean-Charles Fabre, Jean Arlat: Dependability of CORBA Systems: Service Characterization by Fault Injection, SRDS-2002, pp. 276-85, 2002
- R. Chillarege et al. , “Orthogonal Defect Classification — A Concept for In-process Measurements”, IEEE Transactions on Software Engineering, vol.18, n°11, 1992.
- R. Iyer, Z. Kalbarczyck, “Measurement-based Analysis of System Dependability using Fault Injection and Field Failure Data”, Performance 2002, LNCS 2459, pp.290-317, 2002
- D. P. Siewiorek et al. “Reflections on Industry Trends and Experimental Research in Dependability”, IEEE transactions on Dependable and Secure Computing, vol.1, n°2, April-June 2004.
- K. Kanoun, J. Arlat, D. Costa , M. Dalcin, P. Gil, J.-C. Laprie, H. Madeira, N. Suri, “DBench - Dependability Benchmarking”, Supplement of the Int. Conf. on Dependable Systems and Networks, Göteborg, Sweden, 2001, pp. D.12-D.15
- Workshop on Dependability Benchmarking, Supplement Volume of 2002 International Conference on Dependable Systems and Networks (DSN), July 2002, pp. F1-F36, IEEE CS press. Also, papers are available at: http://www.laas.fr/~kanoun/ifip_wg_10_4_sigdeb/external/02-06-25/index.html.

References

Security

- B. Littlewood, S. Brocklehurst, N. Fenton, P. Mellor, S. Page, D. Wright, J. Dobson, J. McDermid and D. Gollmann, “Towards Operational Measures of Computer Security”, *Journal of Computer Security*, 2, pp.211-229, 1993
- M. Dacier, M. Kaâniche, Y. Deswarte, “A Framework for Security Assessment of Insecure Systems”, 1stYear Report of the ESPRIT Basic Research Action 6362: Predictably Dependable Computing Systems (PDCS2), pp. 561-578 , September 1993,
- E. Jonsson and T. Olovsson, “A Quantitative Model of the Security Intrusion Process Based on Attacker Behavior”, *IEEE Transactions on Software Engineering*, 23 (4), pp.235-245, April 1997
- M. Kaâniche, E. Alata, V. Nicomette, Y. Deswarte and M. Dacier, “Empirical Analysis and Statistical Modeling of Attack Processes based on Honeypots”, in *WEEDS 2006 - workshop on empirical evaluation of dependability and security (in conjunction with the international conference on dependable systems and networks, (DSN2006),* pp.119-124, 2006.
- B. B. Madan, K. Goseva-Popstojanova, K. Vaidyanathan and K. Trivedi, “Modeling and Quantification of Security Attributes of Software Systems”, in *IEEE International Conference on Dependable Systems and Networks (DSN 2002),* (Washington, DC, USA), pp.505-514, IEEE computer Society, 2002
- D. M. Nicol, W. H. Sanders and K. S. Trivedi, “Model-based Evaluation: From Dependability to Security”, *IEEE Transactions on Dependable and Secure Computing,*, 1 (1), pp.48-65, 2004.
- R. Ortalo, Y. Deswarte and M. Kaâniche, “Experimenting with Quantitative Evaluation Tools for Monitoring Operational Security”, *IEEE Transactions on Software Engineering*, 25 (5), pp.633-650, 1999
- V. Gupta, V. V. Lam, H. V. Ramasamy, W. H. Sanders and S. Singh, “Dependability and Performance Evaluation of Intrusion Tolerant-Server Architectures”, in *First Latin-American Symposium on Dependable Computing (LADC 2003),* (Sao-Paulo, Brazil), pp.81-101, IEEE Computer Society, 2003
- F. Pouget, M. Dacier, J. Zimmerman, A. Clark, G. Mohay, “Internet attack knowledge discovery via clusters and cliques of attack traces”, Journal of Information Assurance and Security, Volume 1, Issue 1, March 2006 , pp *21-32*
- E. Alata, V. Nicomette, M. Kaâniche, M. Dacier and M. Herrb, “Lessons Learned from the Deployment of a High-Interactiion Honeypot”, in *Sixth European Dependable Computing Conference (EDCC-6),* (Coimbra, Portugal), pp.39-44, IEEE Computer Society, 2006