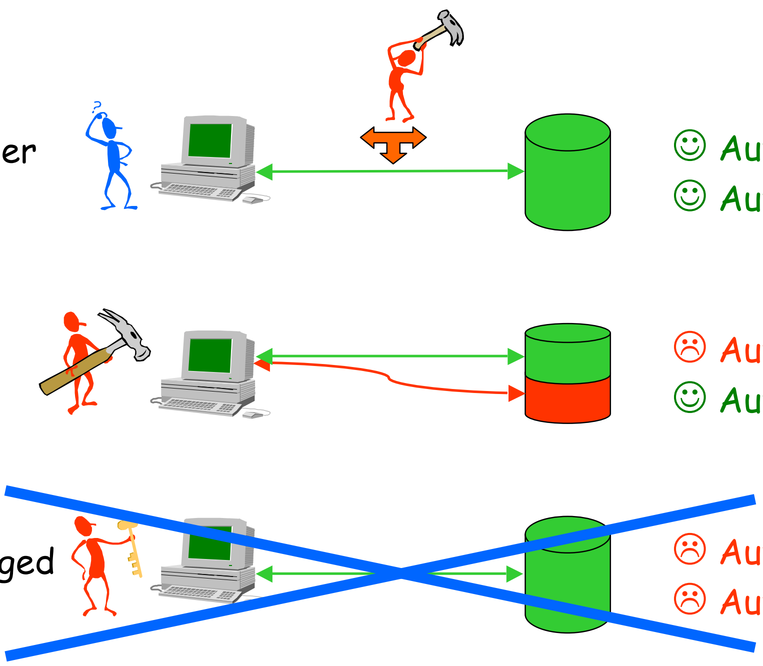
Intrusion Tolerance: Dependability vs. Security

David Powell

Yves Deswarte
LAAS-CNRS
Toulouse, France
deswarte@laas.fr

Who are the attackers/intruders?



1: Outsider — ☺ Authentication ☺ Authorization

2: User — ☹ Authentication ☺ Authorization

3: Privileged User — ☹ Authentication ☹ Authorization

…because the least privilege principle is poorly implemented

"Classical" security

- Authentication

  should prevent non-registered users to access the system

- Authorization

  should prevent users to perform illegitimate actions

… but prevention is insufficient:

Mechanisms are imperfect:

- Authentication can be deceived or bypassed
  - Passwords can be guessed, cracked or disclosed
  - Tokens can be forged, cloned or stolen
  - Biometric sensors can be deceived…

    … and biometric information cannot be revoked!
- Authorization is difficult:
  - Security policy trade-offs: complexity/flexibility
  - Still more difficult for distributed systems
  - Protection mechanisms are inefficient and/or unreliable
- Flaws/malicious logic in implementation

needs more research!

Dependability

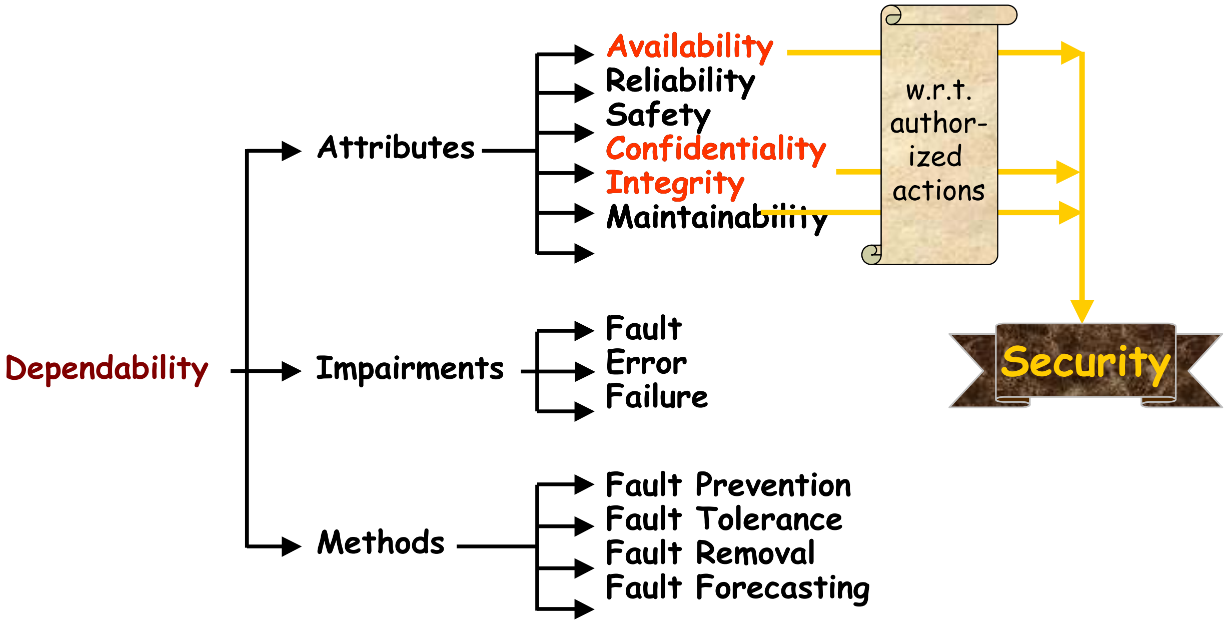
- Trustworthiness of a computer system such that reliance can justifiably be placed on the service it delivers

J.-C. Laprie *(Ed.)*, *Dependability: Basic Concepts and Terminology in English, French, German, Italian and Japanese*, 265p., ISBN 3-211-82296-8, Springer-Verlag, 1992.

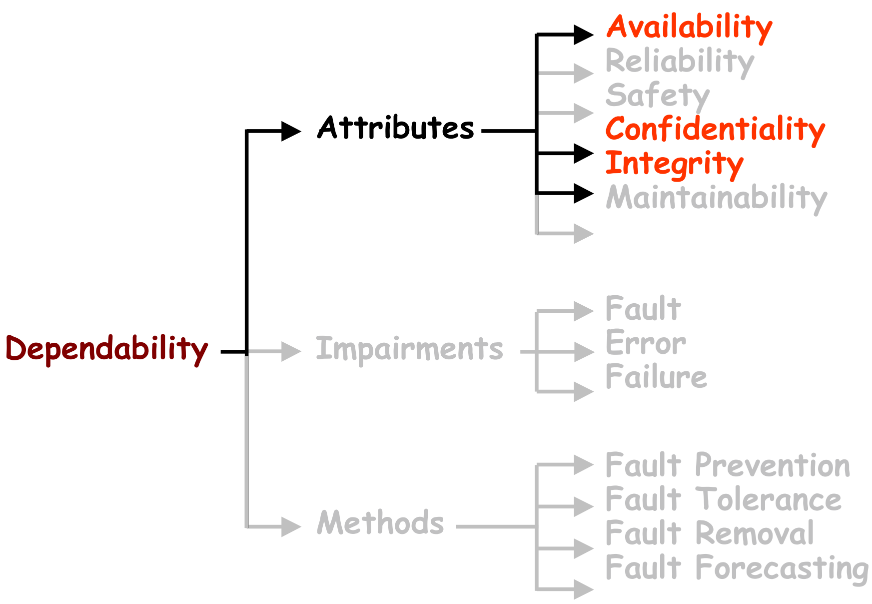
The Dependability Tree

- **Dependability**
  - **Attributes**
    - **Availability**
    - **Reliability**
    - **Safety**
    - **Confidentiality**
    - **Integrity**
    - **Maintainability**
  - **Impairments**
    - **Fault**
    - **Error**
    - **Failure**
  - **Methods**
    - **Fault Prevention**
    - **Fault Tolerance**
    - **Fault Removal**
    - **Fault Forecasting**

The Dependability Tree



Are these attributes sufficient?



Security Properties

Availability
Anonymity
Privacy
Integrity
Authenticity
Secrecy
Confidentiality
Accountability
Non-repudiability
Auditability
Imputability
Irrefutability
Traceability
Opposability

Security Properties

Availability
Anonymity
Privacy
Integrity
Authenticity
Secrecy
Confidentiality
Accountability
Non-repudiability
Auditability
Imputability
Irrefutability
Traceability
Opposability

Security Properties

- Confidentiality
- Integrity
- Availability

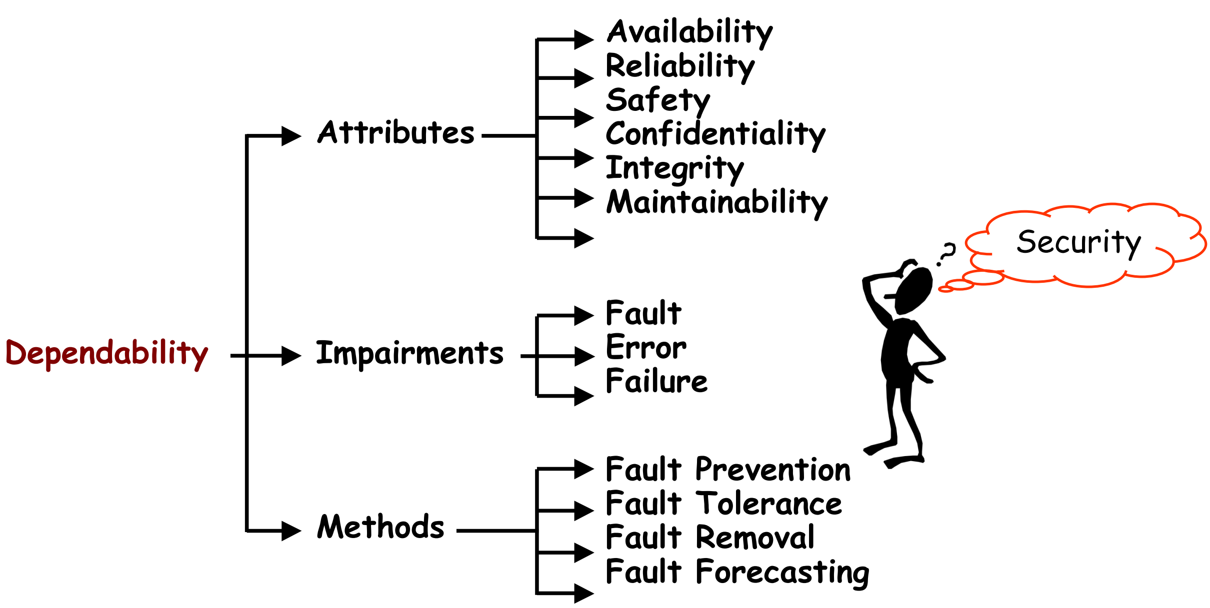
of

- Information
- Meta-information

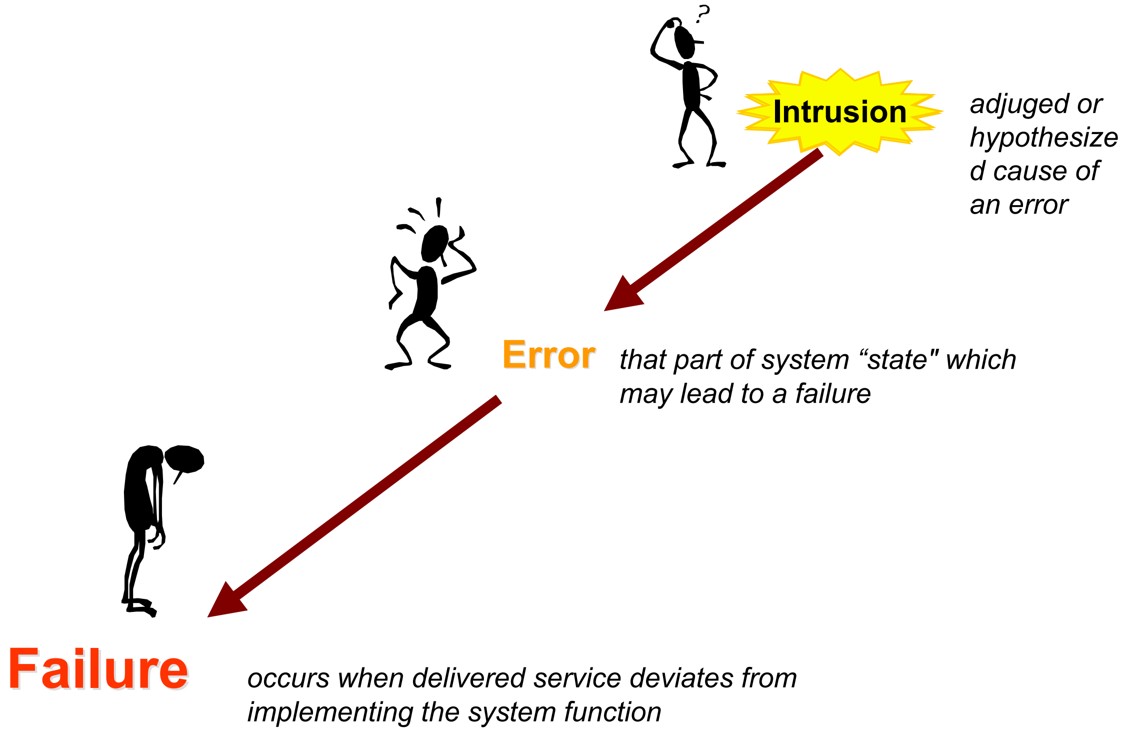
Accountability	*A+I*
Anonymity	*C*
Privacy	*C*
Authenticity	*I*
Non-repudiation	*A+I*

- existence of operation
- identity of person
- personal data
- message content
- message origin
- sender, receiver identity

The Dependability Tree

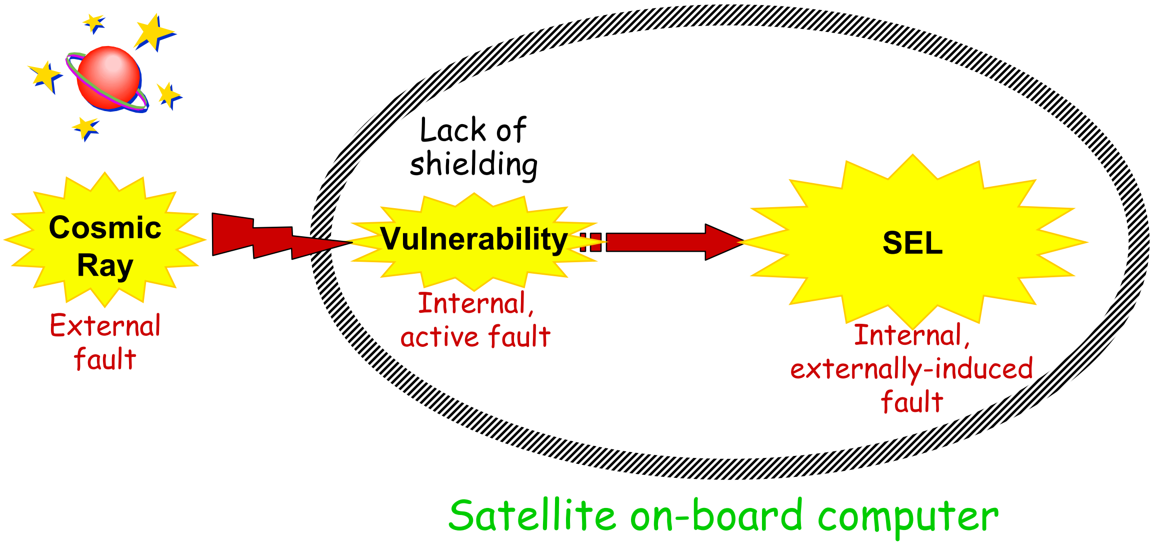


Fault, Error & Failure



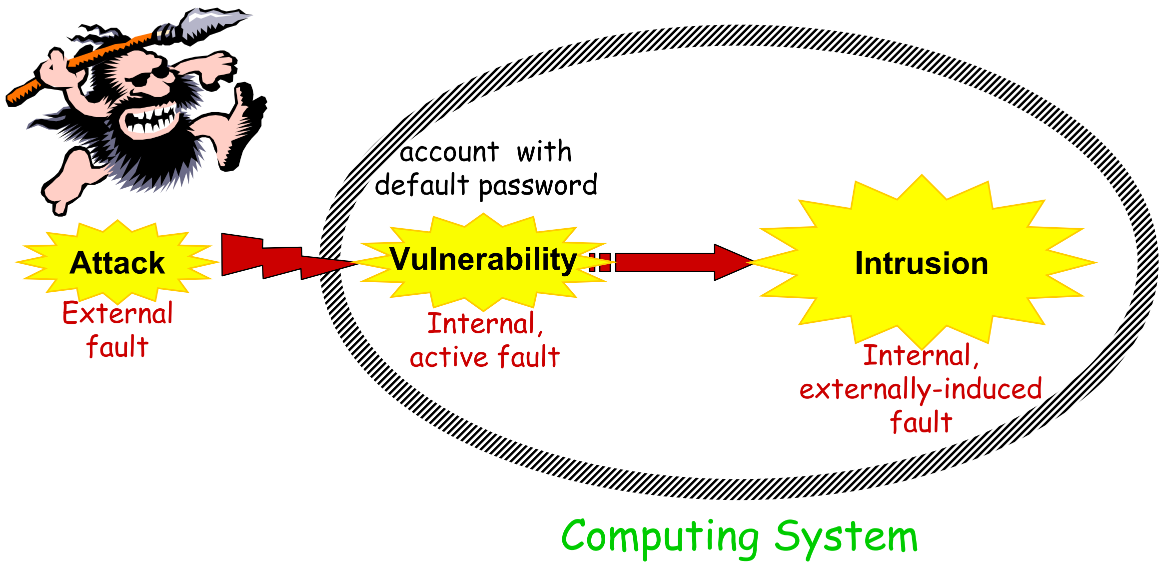
Example: Single Event Latch up

SELs (reversible stuck-at faults)
may occur because of radiation
(e.g., cosmic ray, high energy ions)

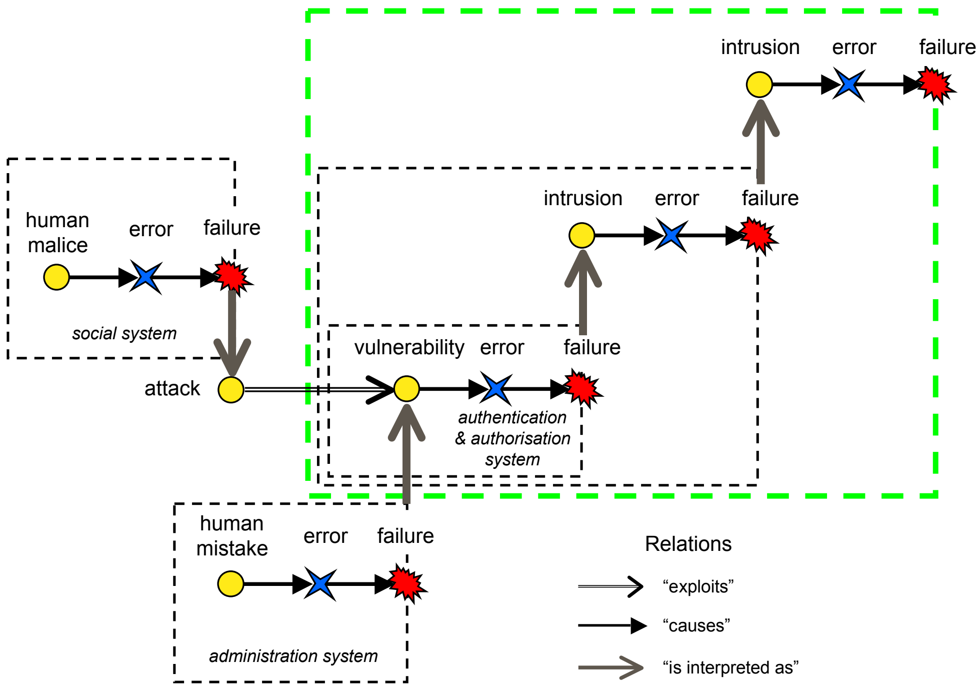


Intrusions

Intrusions result from
(at least partially) successful attacks:



Fault Model

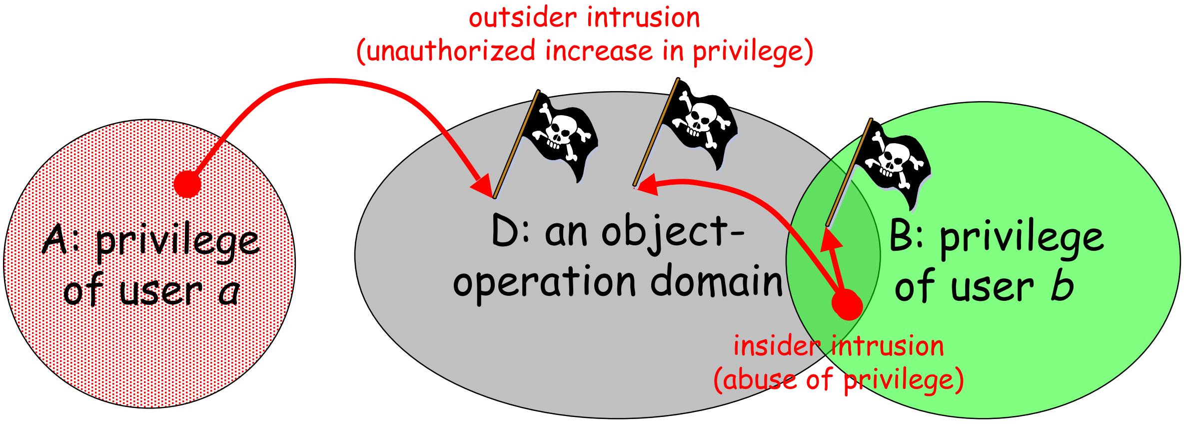


Fault Types

- **attack** - malicious external activity aiming to intentionally violate one or more security properties; an *intrusion* attempt.
- **vulnerability** - an accidental fault, or a malicious or non-malicious intentional fault, in the requirements, the specification, the design or the configuration of the system, or in the way it is used, that could be exploited to create an *intrusion*.
- **intrusion** - a malicious interaction fault resulting from an *attack* that has been successful in exploiting a *vulnerability*.

Outsiders vs Insiders

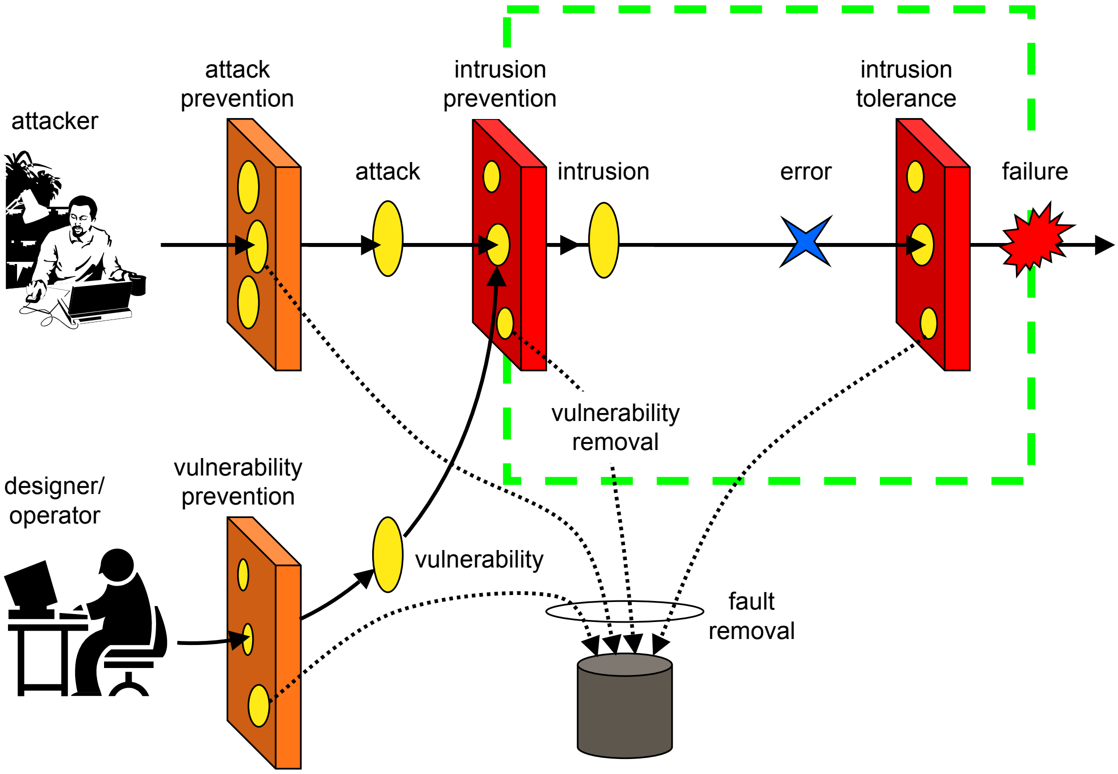
- Outsider: not authorized to perform any of specified object-operations
- Insider: authorized to perform some of specified object-operations



The Dependability Tree

Dependability
- **Attributes**
  - Availability
  - Reliability
  - Safety
  - Confidentiality
  - Integrity
  - Maintainability
- **Impairments**
  - Fault
  - Error
  - Failure
- **Methods**
  - Fault Prevention
  - Fault Tolerance
  - Fault Removal
  - Fault Forecasting

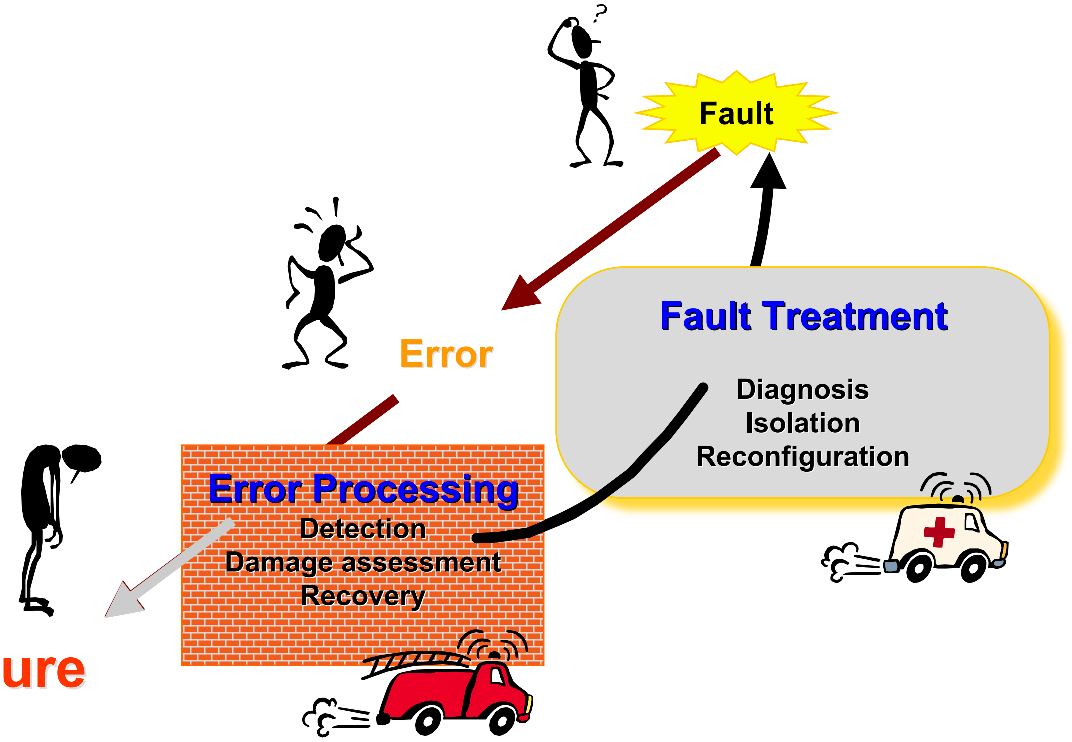
Fault prevention = classical security



The Dependability Tree

- Dependability
  - Attributes
    - Availability
    - Reliability
    - Safety
    - Confidentiality
    - Integrity
    - Maintainability
  - Impairments
    - Fault
    - Error
    - Failure
  - Methods
    - Fault Prevention
    - Fault Tolerance
    - Fault Removal
    - Fault Forecasting

Fault Tolerance



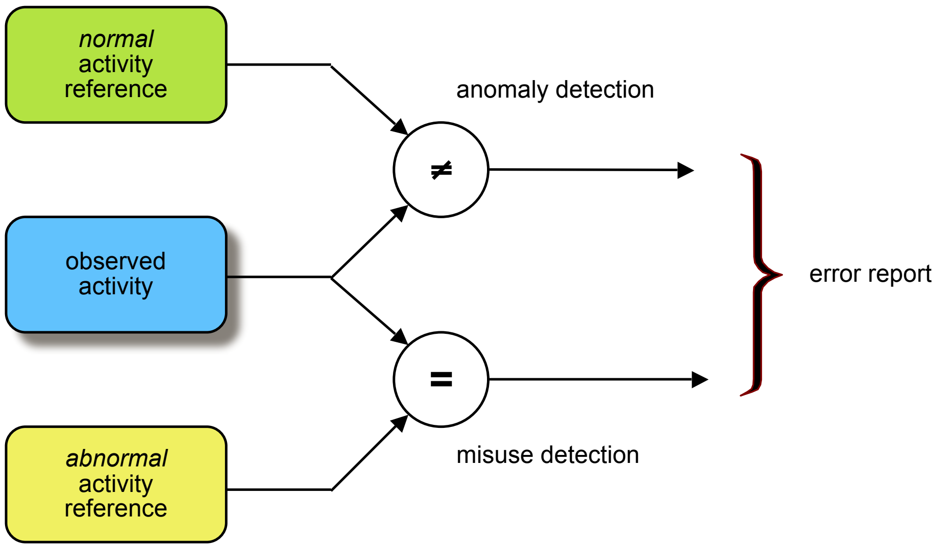
Error Detection (1)

- Likelihood checking
  - by hardware:
    - inexistent or forbidden address, instruction, command…
    - watchdogs
    - error detection code (e.g., parity)
  - by software (OS or application) = verify properties on:
    - values (absolute, relative, intervals)
    - formats and types
    - events (instants, delays, sequences)
  - Signatures (error detection code)

Error Detection (2)

- Comparison between replicates
  - Assumption: a unique fault generates different errors on different replicates
    - internal hardware fault: identical copies
    - external physical fault: "similar" copies
    - design fault / human interaction fault: diversified copies
- On-line model checking

"Intrusion" Detection

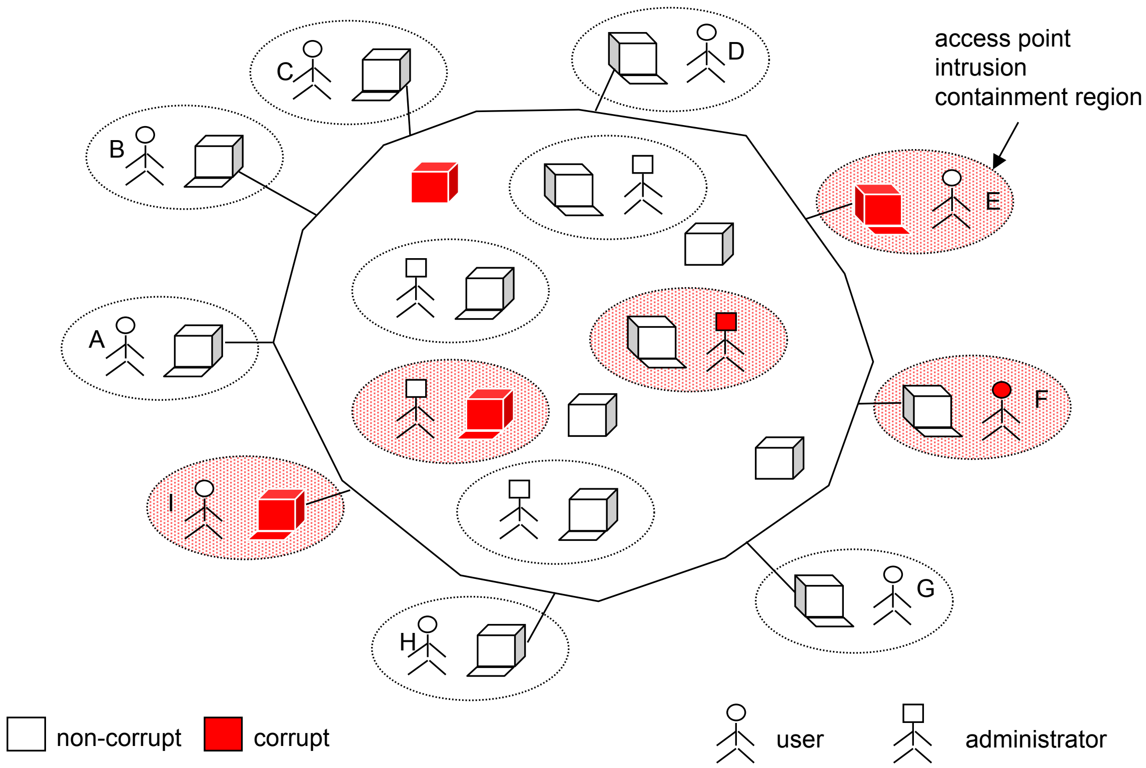


Preemptive Error Detection

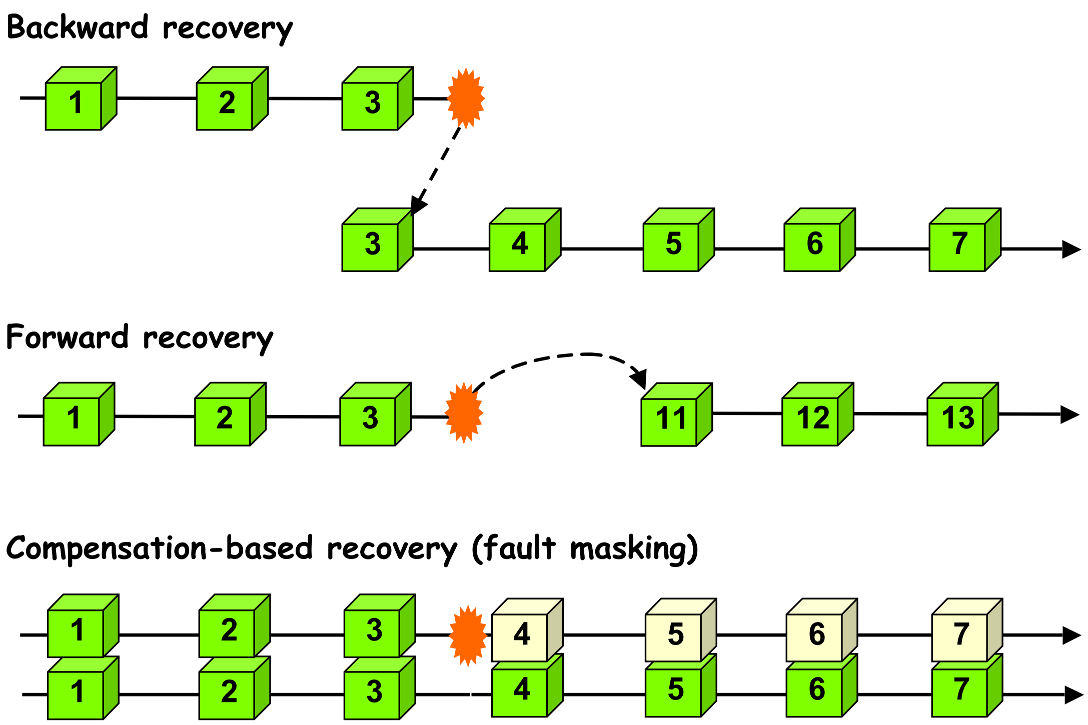
[Avizienis, Laprie & Randell 2001]
(as opposed to concurrent error detection)

- Core concepts: AKA “built-in test”
  - -> Memory scrubbing
  - -> Software rejuvenation
- Interpretation wrt. intrusions
  - Vulnerability scanning
  - Configuration checking

Damage assessment: containment regions



Error Recovery



Error Processing (wrt. intrusions)

- Error detection
  - + Backward recovery (availability, integrity)
  - + Forward recovery (availability, confidentiality)

- Intrusion masking
  - Fragmentation (confidentiality)
  - Redundancy (availability, integrity)
  - Scattering

Intrusion Masking

Intrusion into a part of the system should give access only to non-significant information

FRS: Fragmentation-Redundancy-Scattering

- Fragmentation: split the data into fragments so that isolated fragments contain no significant information: *confidentiality*

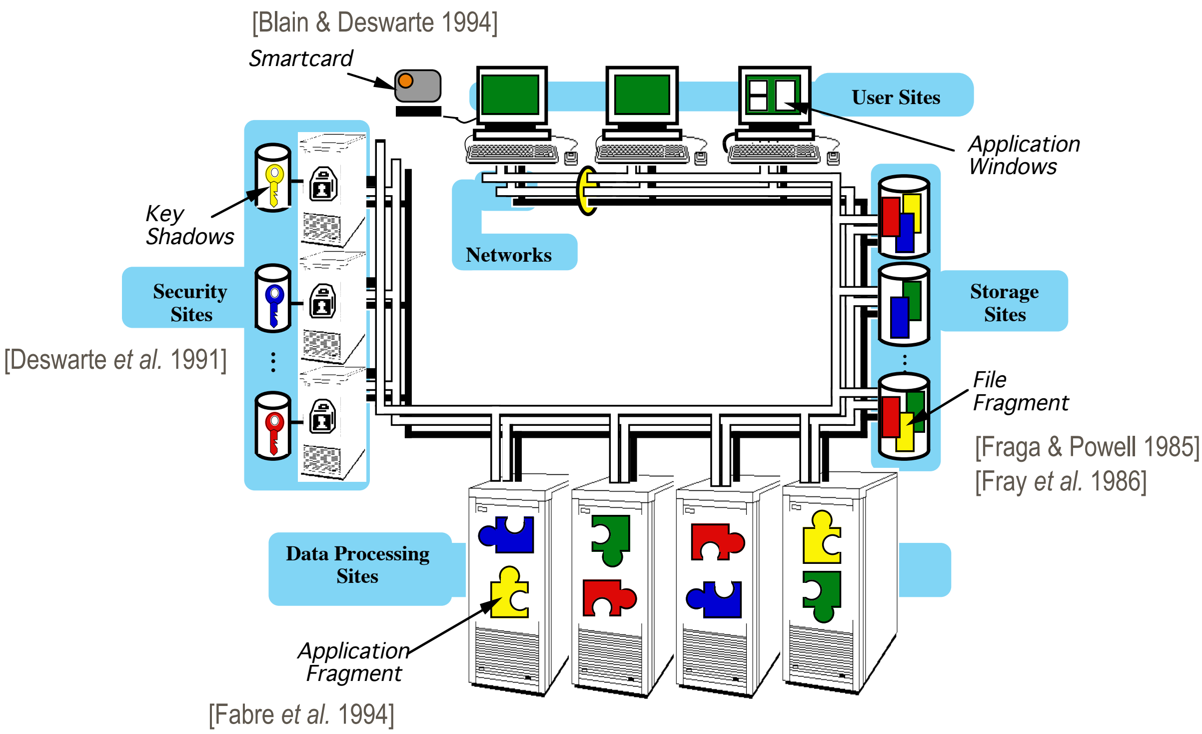
- Redundancy: add redundancy so that fragment modification or destruction would not impede legitimate access: *integrity + availability*

- Scattering: isolate individual fragments

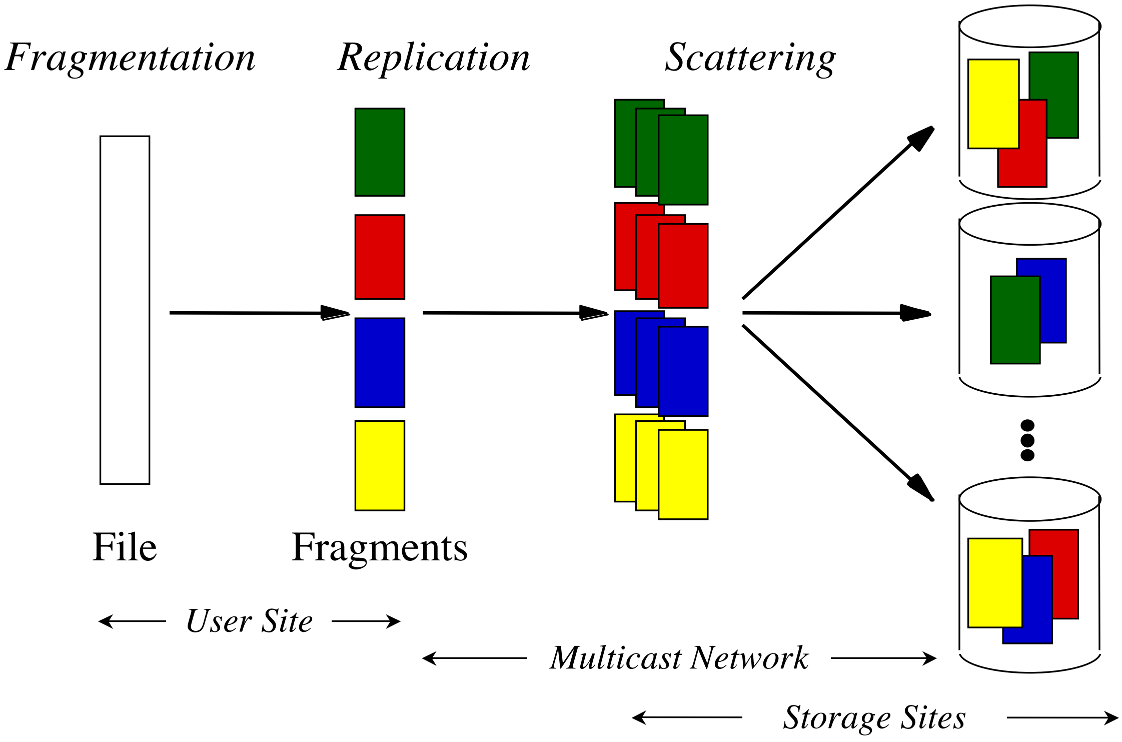
Different kinds of scattering

- **Space**: use different transmission links and different storage sites
- **Time**: mix fragments (from the same source, from different sources, with jamming)
- **Frequency**: use different carrier frequencies (spread-spectrum)
- **Privilege**: require the co-operation of differently privileged entities to realize an operation (separation of duty, secret sharing)

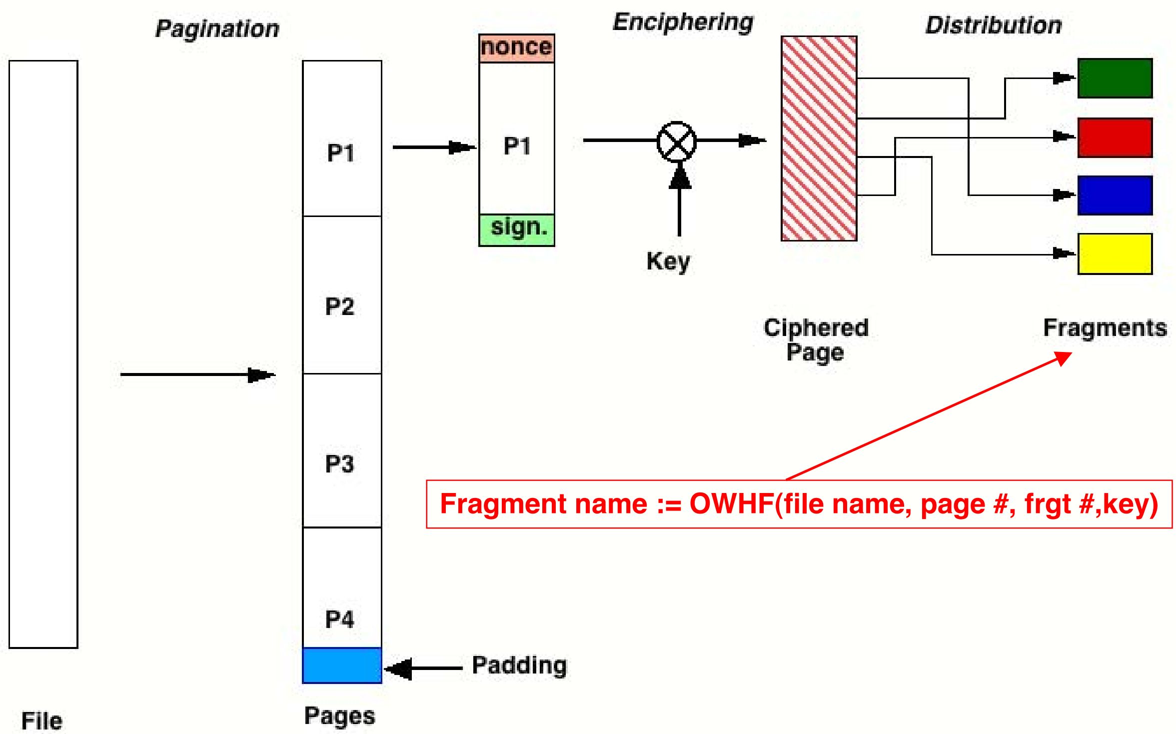
Delta-4 Prototype



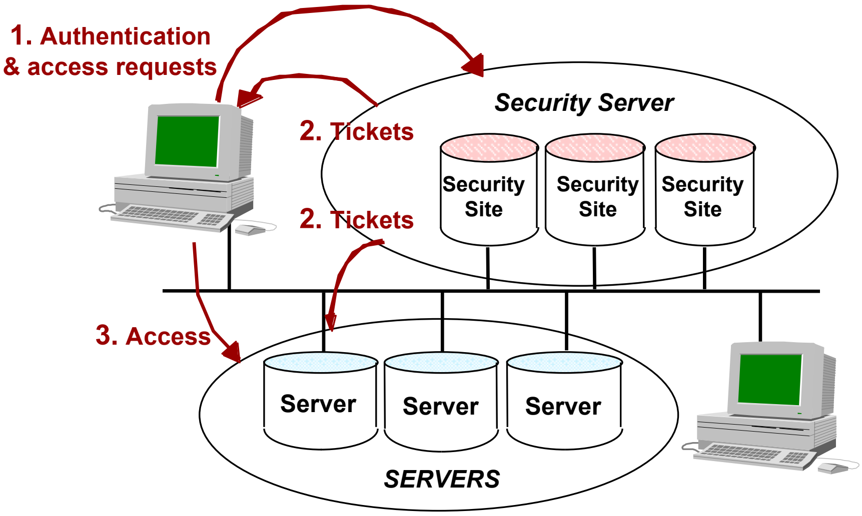
FRSed File Server



File Fragmentation

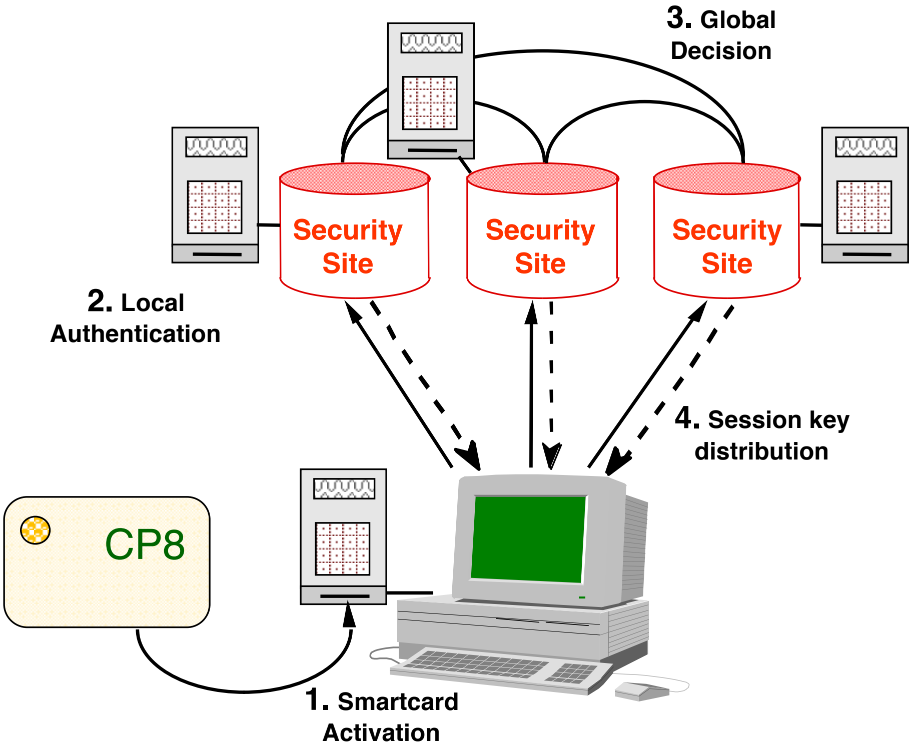


FRSed Security Management

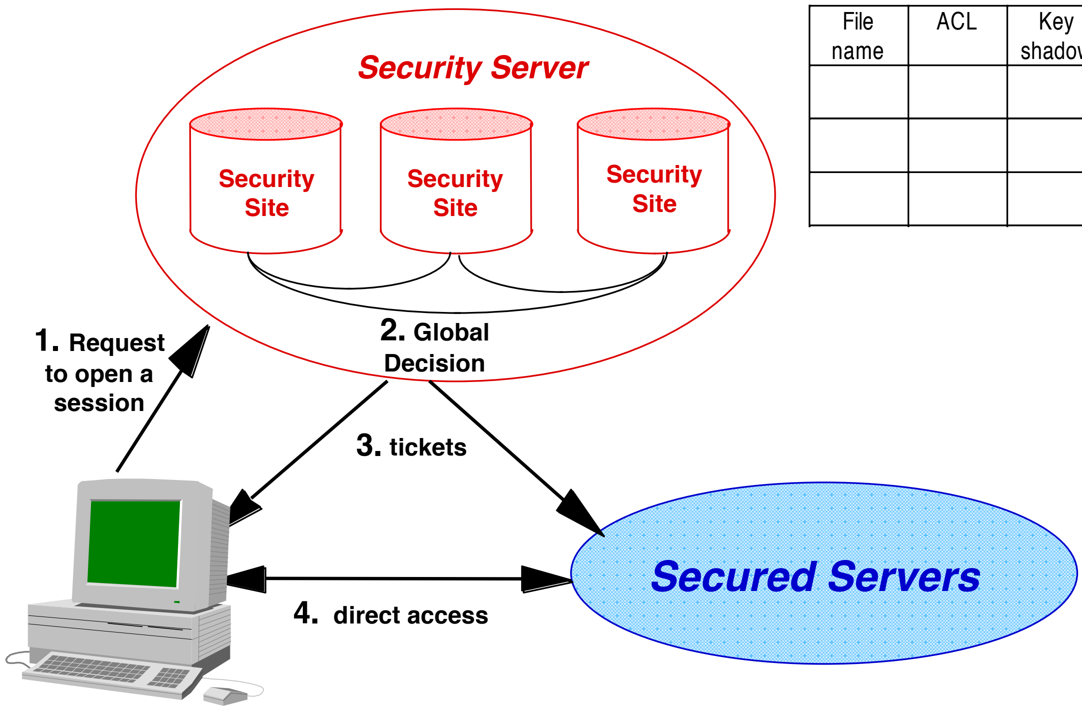


- No single trusted site or administrator
- Global trust in a majority of security sites (and administrators)

Authentication

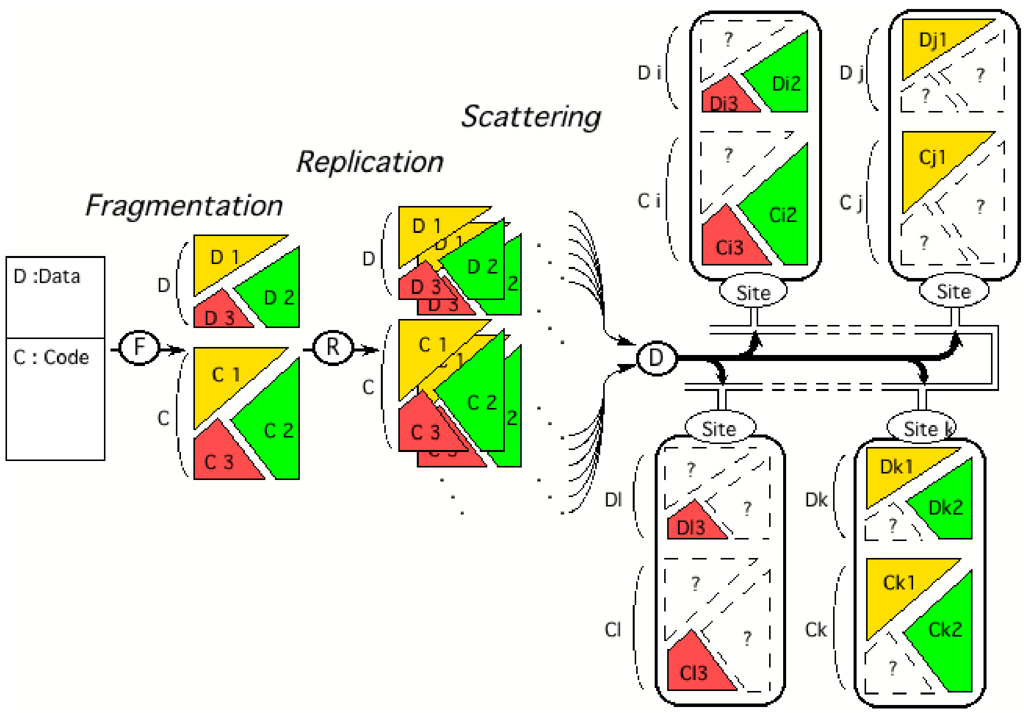


Authorization

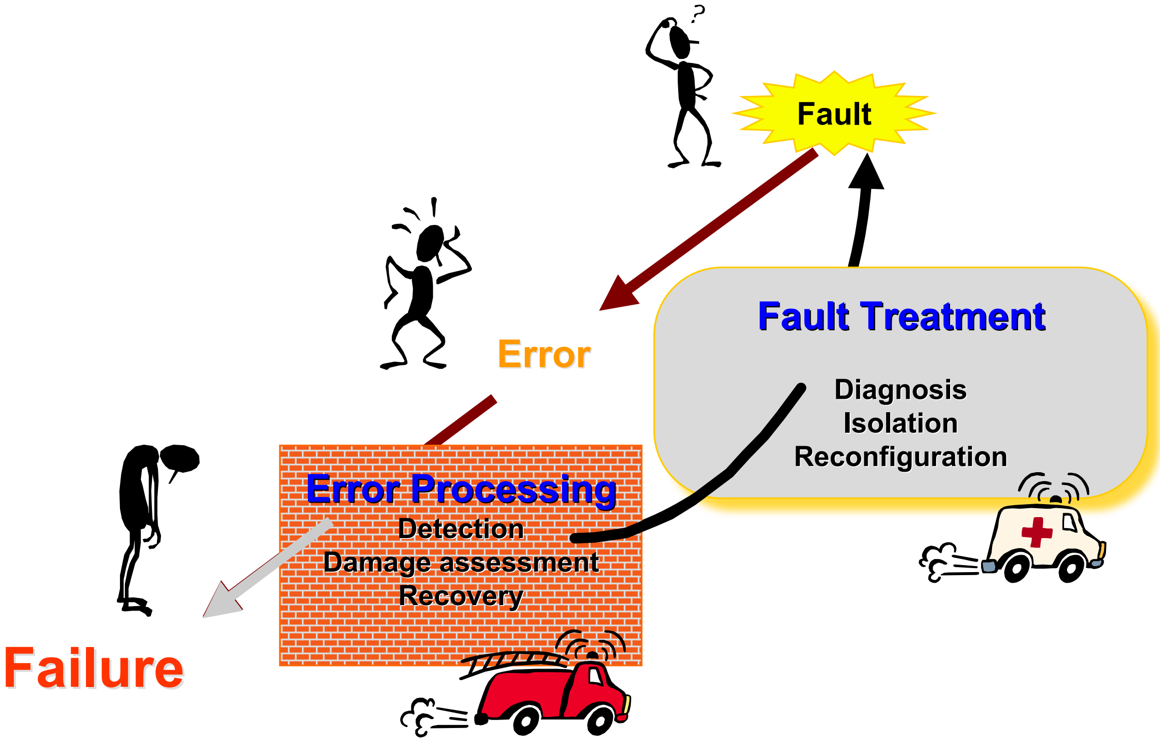


File name	ACL	Key shadow

Fragmented Data Processing



Fault Tolerance



Fault Treatment

- Diagnosis
  - determine cause of error, i.e., the fault(s)
    - localization
    - nature
- Isolation
  - prevent new activation
- Reconfiguration
  - so that fault-free components can provide an adequate, although degraded, service

Fault Diagnosis

- **Intrusion diagnosis**, i.e., trying to assess the degree of success of the intruder in terms of system corruption
- **Vulnerability diagnosis**, i.e., trying to understand the channels through which the intrusion took place so that corrective maintenance can be carried out

  (diagnosis immediate if errors signaled by vulnerability scanner or configuration checker)
- **Attack diagnosis**, i.e., finding out who or what organization is responsible for the attack in order that appropriate litigation or retaliation may be initiated

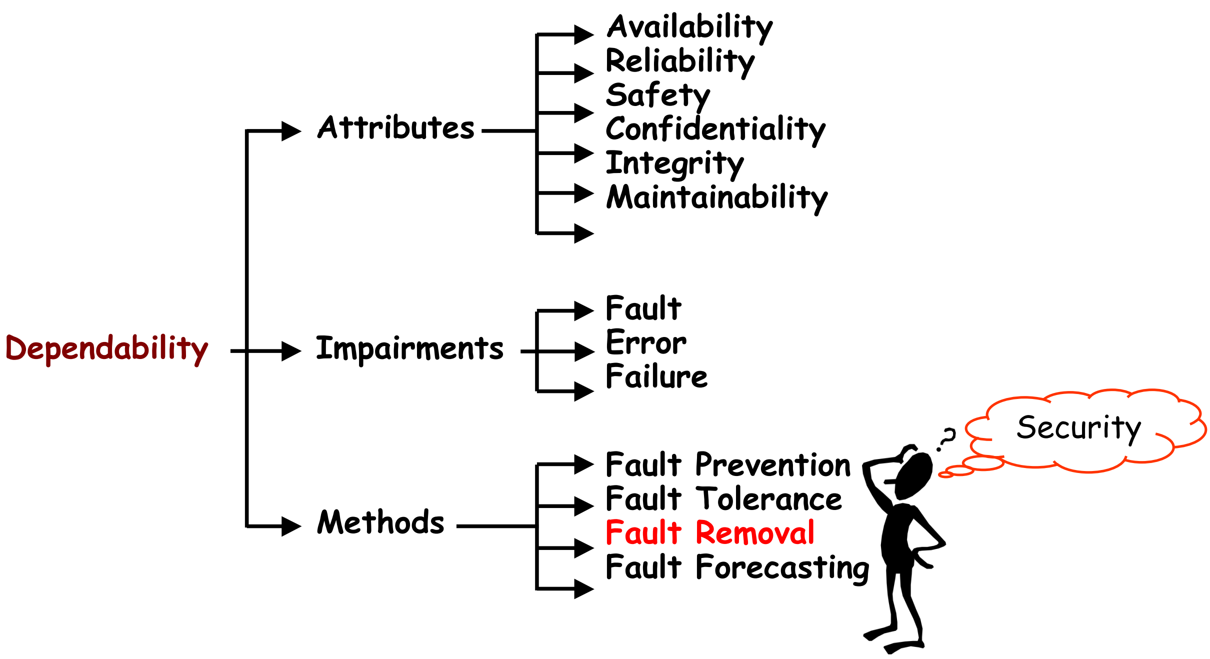
Fault Isolation

- Interpretation wrt. intrusions
  - Blocking traffic from an intrusion containment domain that is diagnosed as corrupt, by, for example, changing the settings of firewalls or routers
  - Removing a corrupted file from the system
- Interpretation wrt. root causes (vulnerability/attack)
  - Taking off line software versions with newly-found vulnerabilities
  - Arresting the attacker

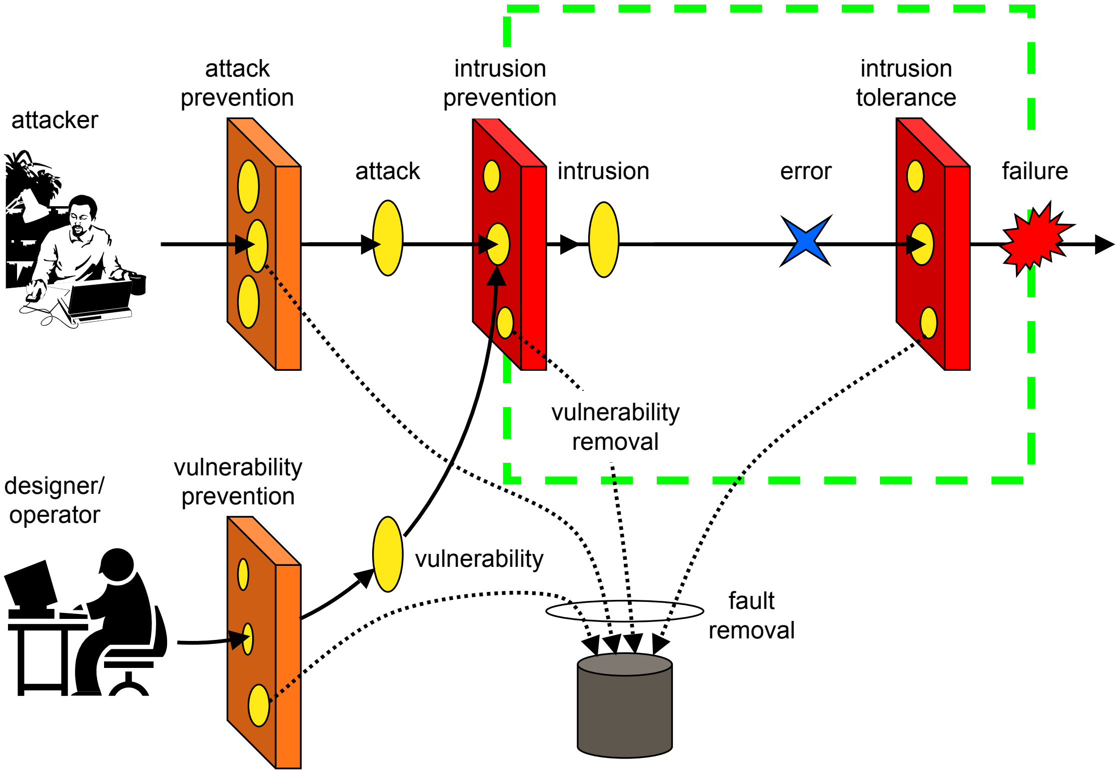
System Reconfiguration

- Interpretation wrt. intrusions
  - Change a voting threshold, e.g., 3/5 => 2/3 after 2 corruptions
  - Deployment of countermeasures, inc. probes and traps
- Corrective maintenance actions
  - Vulnerability removal
    - software revision and upgrade
    - security patches
  - Attacker rehabilitation

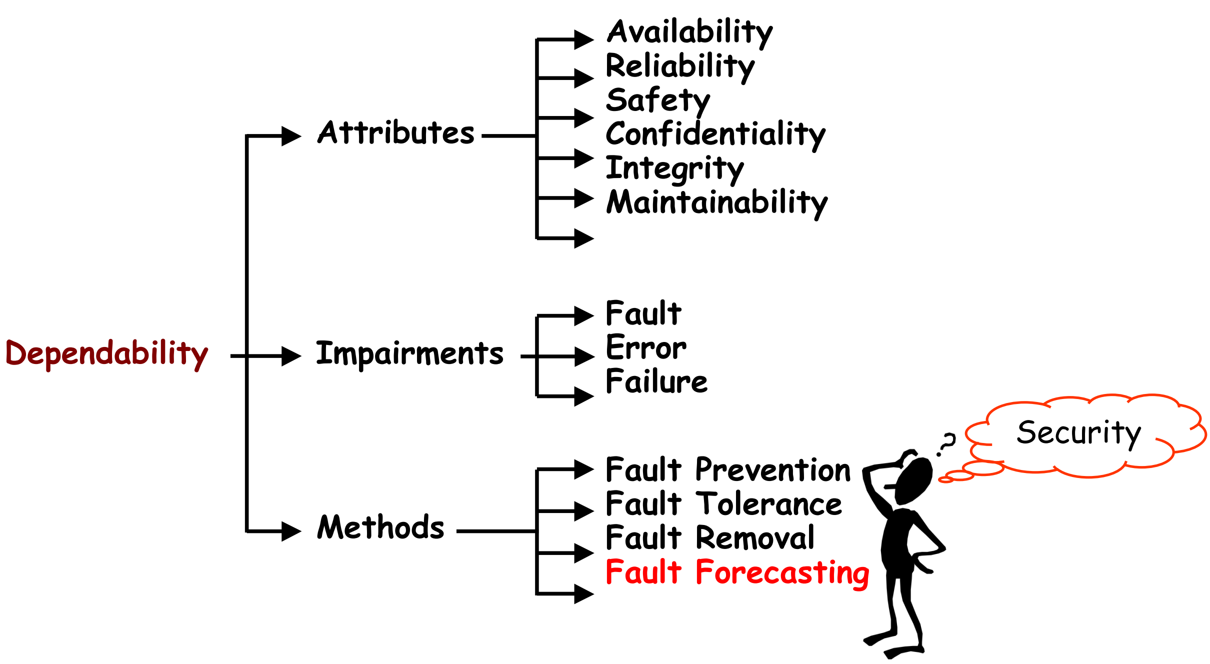
The Dependability Tree



Fault removal



The Dependability Tree



Fault forecasting

- Evaluation:
  - Gain confidence that system dependability is satisfactory
  - Select architecture/components to achieve the best dependability-performance-cost trade-off
- Quantitative measures
  - Reliability: MTFF = mean time to first failure, $R(t) = \text{prob}_{\text{continuous service}}(t)$
  - Availability: MTBF/(MTBF+MTTR), $A(t) = \text{prob}_{\text{correct service provided when needed}}(t)$

Basic assumption

- Faults = elementary component failures (or other rare physical phenomena)
  Model = independent stochastic processes with ~uniform distribution
- OK for physical H/W faults and most environmental faults
- ~OK for most S/W design faults (bugs)
- Not OK for attacks or malicious design faults

Security Evaluation

- Usual techniques
  - Evaluation criteria (TCSEC, ITSEC, CC, ...): ~ qualitative evaluation
  - Risk assessment: subjective evaluation of vulnerabilities, threats, consequences
  - These are static analyses rather than dynamic: "*How the system has been built?*" rather than "*How is it operated?*"

Quantitative security evaluation

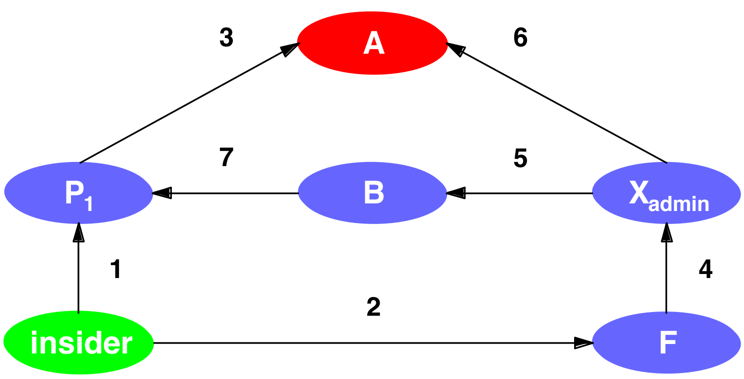
- Measure = effort needed for a possible attacker to defeat the security policy
- Objectives:
  - Take into account security/usability trade-offs
  - Monitor security evolutions according to configuration and use changes
  - Identify the best security improvement for the least usability change

General approach

- Identify security objectives: security policy
- Model (operational) system vulnerabilities
- Model the attack processes
- Compute significant measures

Vulnerability modeling

Privilege graph



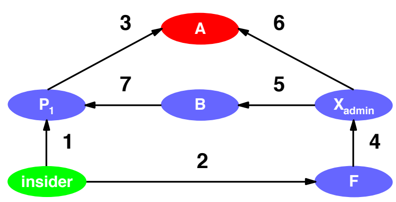
1) X can guess Y's password
2) X can install a Trojan horse that Y can activate
3) X can exploit a flaw in Y's mailer
4) Y is a subset of X
5) Y uses a program that X can modify
6) X can modify a "s-uid" program owned by Y
7) X is in Y's `.rhosts`

- Node = a set of privileges (user, group, rôle, ...)
- Arc = a method to transfer privileges = vulnerability
- Path = set of vulnerabilities usable by a possible attacker to reach a target
- Weight = for each arc, effort to exploit the arc's vulnerability

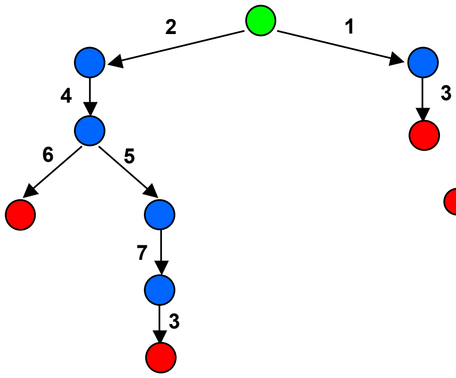
Assumptions on the attack process

- Attack process = all possible successful attack scenarios
- Reasonable assumptions
  - The attacker knows only the vulnerabilities that can be exploited with the privileges he already owns.
  - The attacker will not exploit vulnerabilities which would give him privileges he already owns.
- Plus one out of the two following assumptions:
  - *Total Memory (TM)*: the attacker remembers all the vulnerabilities he did not exploited in the previous steps, and he can "back-track".
  - *Memory-Less (ML)*: the attacker considers only the vulnerabilities that can be exploited with the new privileges he just acquired.

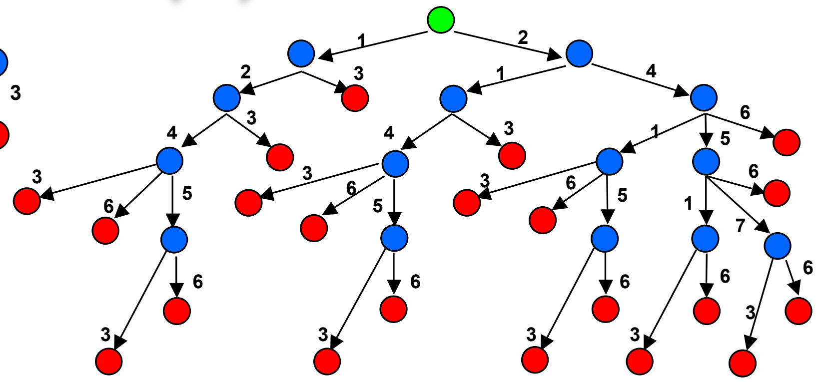
Attack Process Examples



ML Assumption



TM Assumption



Measure computation

① Identify the attacker-target couples

② For each couple, compute:

METF-ML: Mean Effort To security Failure (i.e. to reach the target) with ML assumption.

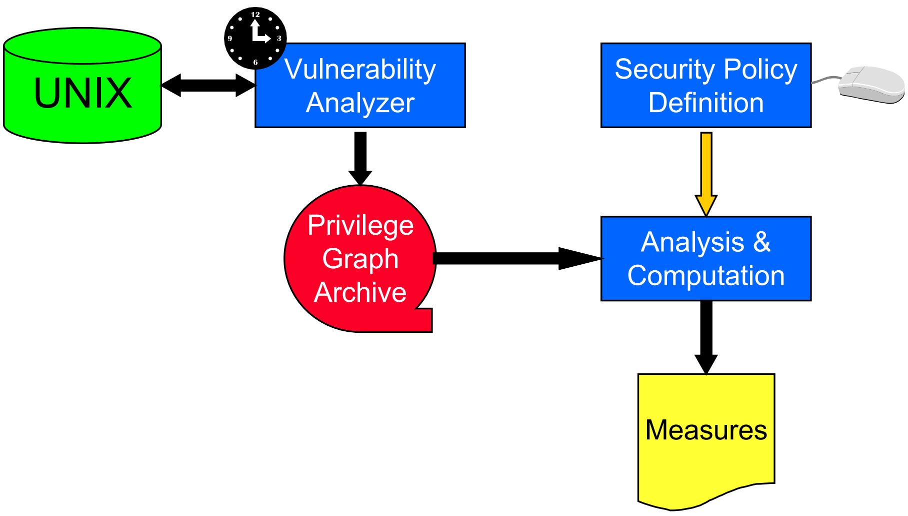
METF-TM: Mean Effort To security Failure with TM assumption.

Shortest Path: Mean effort to go through the shortest path.

Number of Paths: Number of possible paths from the attacker to the target nodes.

ESOPE Tool Set

(Évaluation de la Sécurité OPErationnelle)



Experiment report

- Objectives:
  - Validate the approach:
    - Assess the measure pertinence wrt. system changes (configuration, users, …)
    - Feasibility of a full-size system evaluation.
  - *Was not aimed*:
    - Correct the identified vulnerabilities

Experiment context

Target system:

- Unix
- 700 users - 300 machines - LAN
- 13 months (June 1995 - July 1996)

13 types of vulnerabilities
(files `.rhosts`, `.*rc`, passwords, etc.)

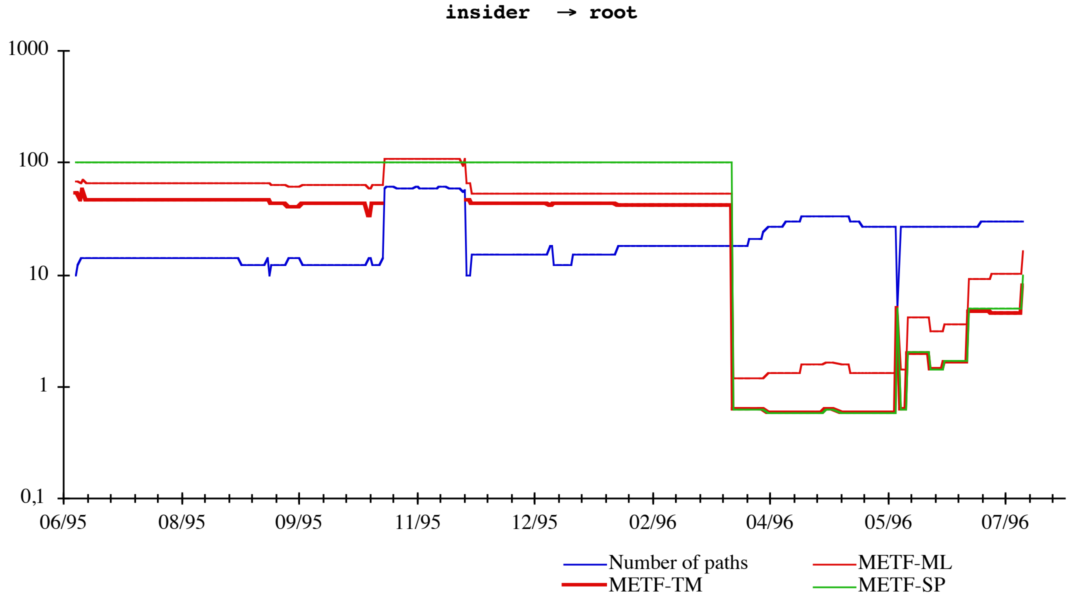
4 difficulty levels:

Type	Weight
immediate	10
easy	10^2
difficult	10^3
very difficult	10^4

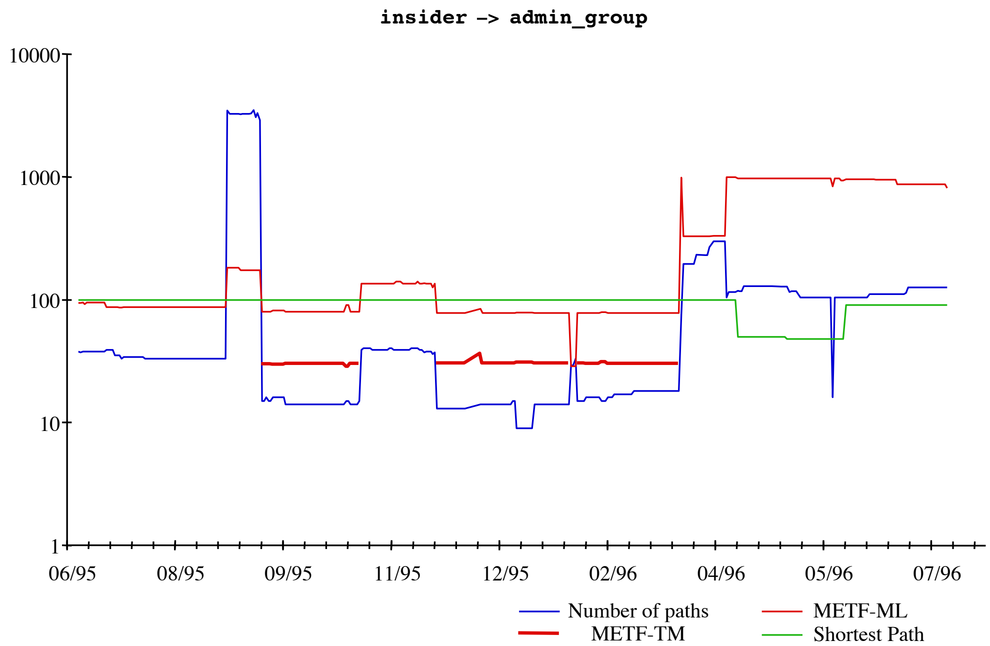
Security objectives:

	Attacker	Target
Objective 1	insider	root
Objective 2	insider	admin_group

Results (1)



Results (2)



Comparison between measures

- The shortest path (**SP**) is not sensitive enough to identify important events
- The number of paths (**NP**) changes too often and would produce a large number of false alarms.
- **METF-ML** presents a good sensitivity to important events.
- **METF-TM** is easier to interpret, but is sometimes too complex to be computed.

References

- A. Avizienis, J.-C. Laprie, B. Randell, *Fundamental Concepts of Dependability*, LAAS Report N°01145, April 2001, 19 pp.
- Y. Deswarte, L. Blain and J.-C. Fabre, "Intrusion Tolerance in Distributed Systems", in *IEEE Symp. on Research in Security and Privacy,* Oakland, CA, USA, 1991, pp.110-121.
- J.-C. Laprie (Ed.), *Dependability: Basic Concepts and Terminology in English, French, German, Italian and Japanese*, 265p., ISBN 3-211-82296-8, Springer-Verlag, 1992.
- R. Ortalo, Y. Deswarte, M. Kaâniche, "Experimenting with Quantitative Evaluation Tools for Monitoring Operational Security", *IEEE Transactions on Software Engineering*, Vol.25, N°5, pp.633-650, Sept./Oct. 1999.
- D. Powell, A. Adelsbasch, C. Cachin, S. Creese, M. Dacier, Y. Deswarte, T. McCutcheon, N. Neves, B. Pfitzmann, B. Randell, R. Stroud, P. Veríssimo, M. Waidner, "MAFTIA (Malicious- and Accidental-Fault Tolerance for Internet Applications)", *Sup. of the 2001 International Conference on Dependable Systems and Networks (DSN2001)*, Göteborg (Sweden), 1-4 July 2001, IEEE, pp. D32-D35.