Challenges for Providing Processing Integrity in Grid Computing

Felipe Martins[1], Márcio Maia[1], Rossana M. de C. Andrade[2]
Aldri L. dos Santos[2], José Neuman de Souza[1,2]

[1] Teleinformatics Engineering Department - Federal University of Ceará
[2] Computer Science Department – Federal University of Ceará

{felipe,marcio}@cenapadne.br,
{rossana,aldri,neuman}@lia.ufc.br

GREaT

CENAPAD-NE

Schedule

- Computational Grids
- Grid Security Attacks
- Classification of Misbehavior Faults
- Treatment of Malicious Faults
- System-Level Diagnosis
- Diagnosis Applied to Grids
- Grid Simulators
- Case Study
- Final Remarks

GREaT

CENAPAD-NE

Computational Grids

- Gathering, selection and sharing of distributed resources
  - Heterogeneity
  - Geographic dispersion
  - Transparent access to the resources
- More complex security requirements
- Grids are more susceptible to security attacks
  - User and servers masquerading
  - Abusive usage of the resources
  - Non-authorized access to the services
  - Subversion of the resources

Attacks against Grids

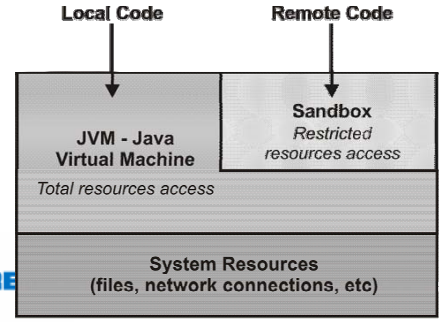
- Threats to the *dependability*
  - DoS (Denial-of-Service)
    - Defense → access control
      - Inefficient against internal attacks
  - DoS or DDoS (Distributed DoS) used into the grid itself or against another grid site
    - Defense → limitation of the resources usage

Attacks against Grids

- Threats to the *privacy*
  - User masquerading or eavesdropping
  - Searching for temporary files
  - Defense → cryptographic keys and SSL tunnel

Attacks against Integrity in Grids

- Protecting the Resources
  - To ensure the environment is not " "contaminated" with malicious codes
  - To encourage a greater participation and availability
  - Viruses, worms, trojans
  - Defense → virtualization



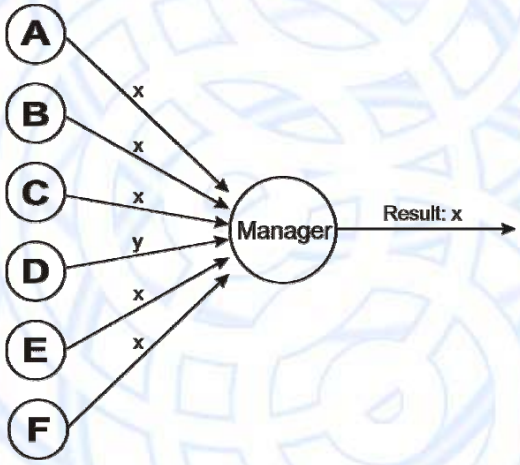
- Protecting the Applications
  - To ensure the environment is not "contaminated" with malicious hosts
  - Applications endangered by incorrect results
  - Non-trivial task
    - Data Transmission
    - Job Processing

Classification of Misbehavior Faults

- Inactive nodes
  - Do not cooperate to the network
  - Avoid forwarding packets
  - Refuse to process the jobs
  - Omit information about available resources
- Selfish nodes
  - Neglect help to other nodes
  - OurGrid
    - Free-rider
    - Consume resources from the grid without providing its own resources once requested
- Malicious nodes
  - Subvert the grid resources
  - Provide an invalid result
  - Spread viruses and worms

Treatment of Malicious Faults

- Fault Tolerance Common Techniques
  - Majority Voting
    - Jobs replicas are distributed among the nodes
    - Majority of results matching is taken as valid
  - Spot-Checking
    - Test jobs whose results are previously known
    - Blacklist

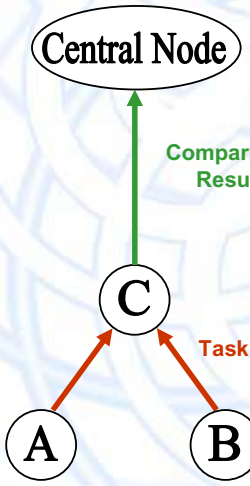


Treatment of Malicious Faults

- Reputation
  - Nodes with good reputation → better resource providers
    - Nodes do not need to be tested so frequently
    - It reduces the processing *overhead*
  - Highly used in P2P systems
    - File sharing
    - Minimize the presence of peers interested in diffusing false or incomplete files, and also viruses and worms

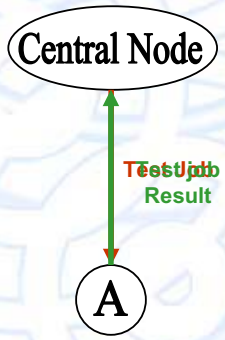
System-Level Diagnosis

- Strategy of fault tolerance
- Sequence of tests
  - Which units are faulty and which are fully functional
  - Syndrome = set of obtained results
- Diagnosis Models
  - PMC, ADSD, Hi-ADSD
  - Comparison-based
    - MM, Broadcast, and others



Diagnosis Applied to Grids

- Defense against manipulation attacks
  - Considers the heterogeneous and dynamic nature of such environments
  - Public and private grids
- Proposed Solution
  - Diagnosis combined to spot checking and reputation
- Remarks
  - Tests Format
    - Different non-faulty nodes (non-malicious) may provide different results to a same task
  - Time to answer a test
    - Round test time is limited
    - Nodes with different processing capacities lead to different response times
    - Highly dispersed (intercontinental grid)



Grid Simulators

- OptorSim, GridNet, MicroGrid, SimGrid and GridSim

Feature	OptorSim	GridNet	MicroGrid	SimGrid	GridSim
Using	Simulator	Simulator	Emulator	Simulator	Simulator
Language	Java	C++	C	C	Java
Manual	Good	Poor	Good	Very Good	Very Good
Portability	Yes	No	No	No	Yes
Extensibility	Good	High	Low	Regular	Good
Engine	Multithreads	Serial	Parallel	Serial	Multithreads

Case Study

- Simulations
  - GridSim 3.3
  - New introduced methods
  - Without reputation scheme
- Scenarios
  - 10.000 *jobs*
  - 200 worker nodes
  - Percentage of malicious nodes
    - 1/6, 1/3 and 2/3 of the grid nodes providing bad results
  - Amount of test rounds
    - 3, 5, 8, 10, 15 and 20

Case Study

- Metrics
  - Amount of necessary test rounds
  - Overhead
  - Impact of the blacklist
- Not all jobs are corrupted by the malicious nodes
  - Probability of 25% chances of returning an invalid result
- Node with more than 3% of errors ⇒ blacklist
- Each experiment, 100 simulation runs

Test Jobs

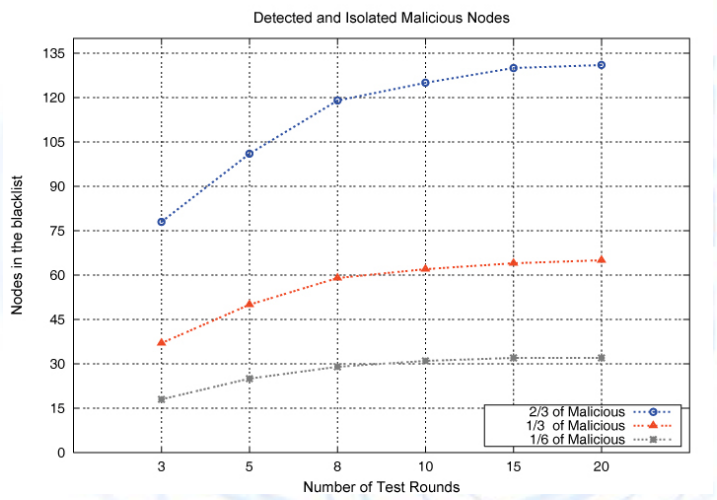
- Factoring of a string randomly generated
- ASCII code of each character is multiplied by an element from a finite set of prime numbers
- Result is the sum of all factors multiplication
- Example
  - String “abcde”
  - Set of primes {3,5,7,11}
  - Result: 97 x 3 + 98 x 5 + 99 x 7 + 100 x 11 + 101 x 3 = **2877**

GREAT

CENAPAD-NE

Detected Malicious Nodes

- Practically all malicious nodes are detected with 15 test rounds
- More that 20 rounds the benefit is insignificant

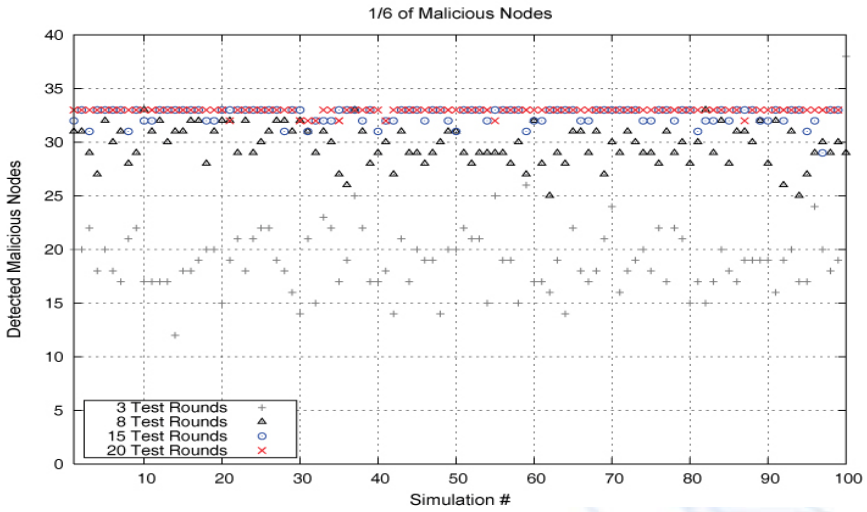


GREAT

CENAPAD-NE

Detected Malicious Nodes

- 15 test rounds offer an effectiveness similar to 20 test rounds
- Scheme is unstable with just 3 rounds
  - In the best case, 26 detected nodes
  - In the worst, only 12 detected nodes



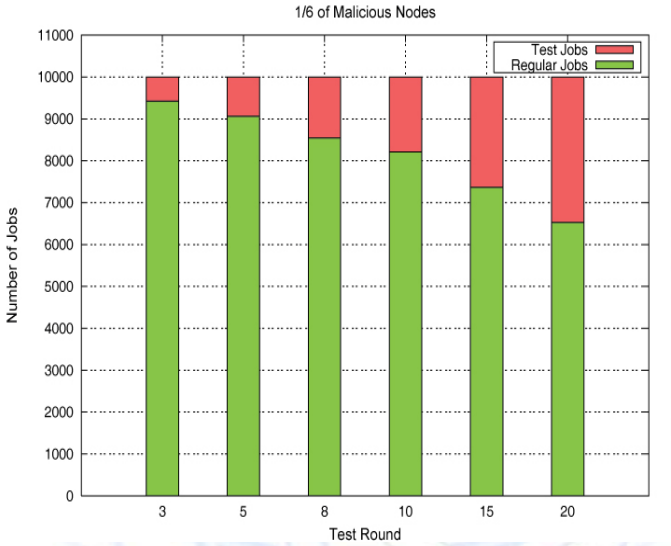
Detected Malicious Nodes

- Spot-checking and blacklist are inefficient with just 3 rounds
- Better results after 8 rounds
- The worst case percentage rises as the number of malicious nodes increases
- The higher the number of test rounds and malicious nodes, lower the variance

Malicious Nodes	3 Rounds		8 Rounds		15 Rounds	
	Worst Case	Best Case	Worst Case	Best Case	Worst Case	Best Case
1/6 (33 nodes)	37 %	79 %	75 %	100 %	87 %	100 %
1/3 (66 nodes)	42,5 %	73 %	80 %	98,5 %	92,5 %	100 %
2/3 (133 nodes)	51 %	67 %	83,5 %	94,5 %	95,5 %	100 %

Overhead

- 15 test rounds
  - High overhead
  - From 10.000 jobs, over 2.500 are just for test
- 8 test rounds
  - Acceptable trade-off
  - With 1/6 of malicious nodes, 30 from 33 were detected
- Reputation can reduce even more overhead

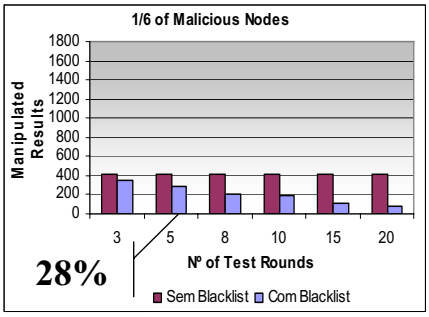


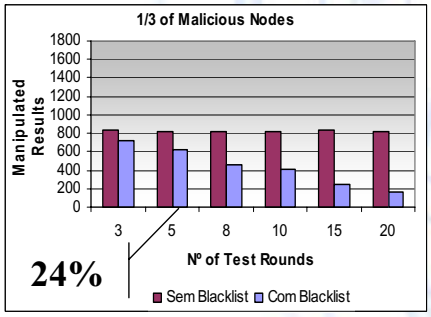
GREaT

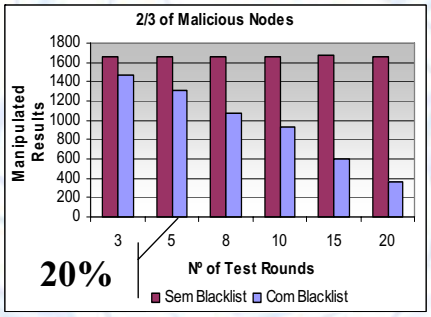
CENAPAD-NE

Blacklist

- Without blacklist
  - Number of manipulated results remains the same
  - Double the number of malicious nodes, double the manipulated results
- With blacklist
  - Manipulated results decrease with more test rounds
  - Less efficiency with a higher number of malicious nodes
  - Example: Manipulated results with 5 test rounds







GREaT

CENAPAD-NE

Final Remarks

- Nowadays, no existing grid platform presents security mechanisms for processing integrity
- Presence of malicious nodes can be detected and minimized with fault tolerance techniques
- A reputation scheme with blacklist can increase security in the environment

Final Remarks

- A possible and efficient scalable approach
  - Apply these concepts in a diagnosis model
  - Even with different quota of malicious nodes, practically all can be detected and isolated
- Future work
  - A further study to use a reputation scheme
  - Scrutinize other possible metrics and scenarios
    - Treat other kinds of misbehavior nodes
  - Investigate the usage of this solution in real grids
    - OurGrid and Globus

Questions?

Felipe Sampaio Martins
felipe@cenapadne.br