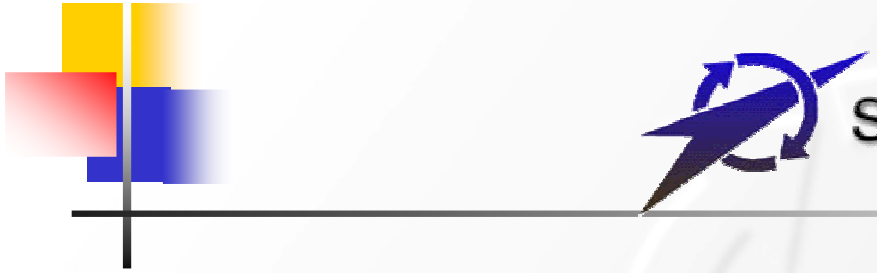


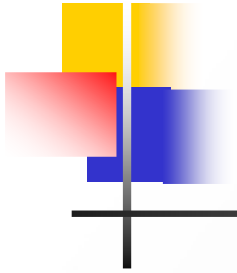
Scalability and performance Analysis of Tamper Resistance Methods in RE-TRUST

Vasily Desnitsky, Igor Kotenko

Computer Security Research Group,
St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences

Agenda

- Performance & Scalability
- TR methods
- Definition of Scalability and Security
- Optimizing of TR methods
- TR methods choosing
- Performance & Scalability Analysis
- Performance metrics
- TR methods performance determination
- Security policies
- Future work



Performance & Scalability

- The problem of performance and scalability
- Protection mechanism implementation in practice
- Minimizing of trusted server side computations
- Complexity of TR-methods support and verifications on the trusted server
- Some possible tradeoff achieving
  - Security quality vs. Scalability
- Overall security level provided by the protection mechanism

TR methods (1/2)

- RE-TRUST solutions classified as *Remote* ones
  - Barrier Slicing (BS)
  - Barrier Slicing with tamper resistant hardware
  - Barrier Slicing with trade-off
  - Continuous Replacement
  - Orthogonal Replacement (OR)
  - Secure interlocking and authenticity checking
  - Control Flow Checking (CFC)
  - Invariant Checking (IC)
  - Hardware assisted invariants monitoring
  - Remote Attestation with TPM
  - Monitor that performs Checksums on a program
  - First Torino prototypes using AOP and JVMTI

TR methods (2/2)

- The aim is
  - To analyze each TR method for the purpose for estimation of it's resources consumption
    - Central processor resources
    - Memory amount required by the method
  - To determine *security power* of each method
    - Some relative notion reflecting the force of each method and hence the importance of it's application in practice
    - Weaker goal: To determine just the priority all the methods to be applied when there are essential constraints of system resources
  - To define strategies of choosing concrete methods to implement

Scalability and Security definitions

- Scalability
  - By the *scalability aim* of the mechanism we mean the dependency of its server's side computational complexity upon the client amount being executed simultaneously is close to the linear one
  - Certainly the linear dependency is not achievable here
- Security provided by a TR method
  - Open question: what does security exactly mean and how to measure it?
  - Give some relative weight to each method, that could be determined by experts

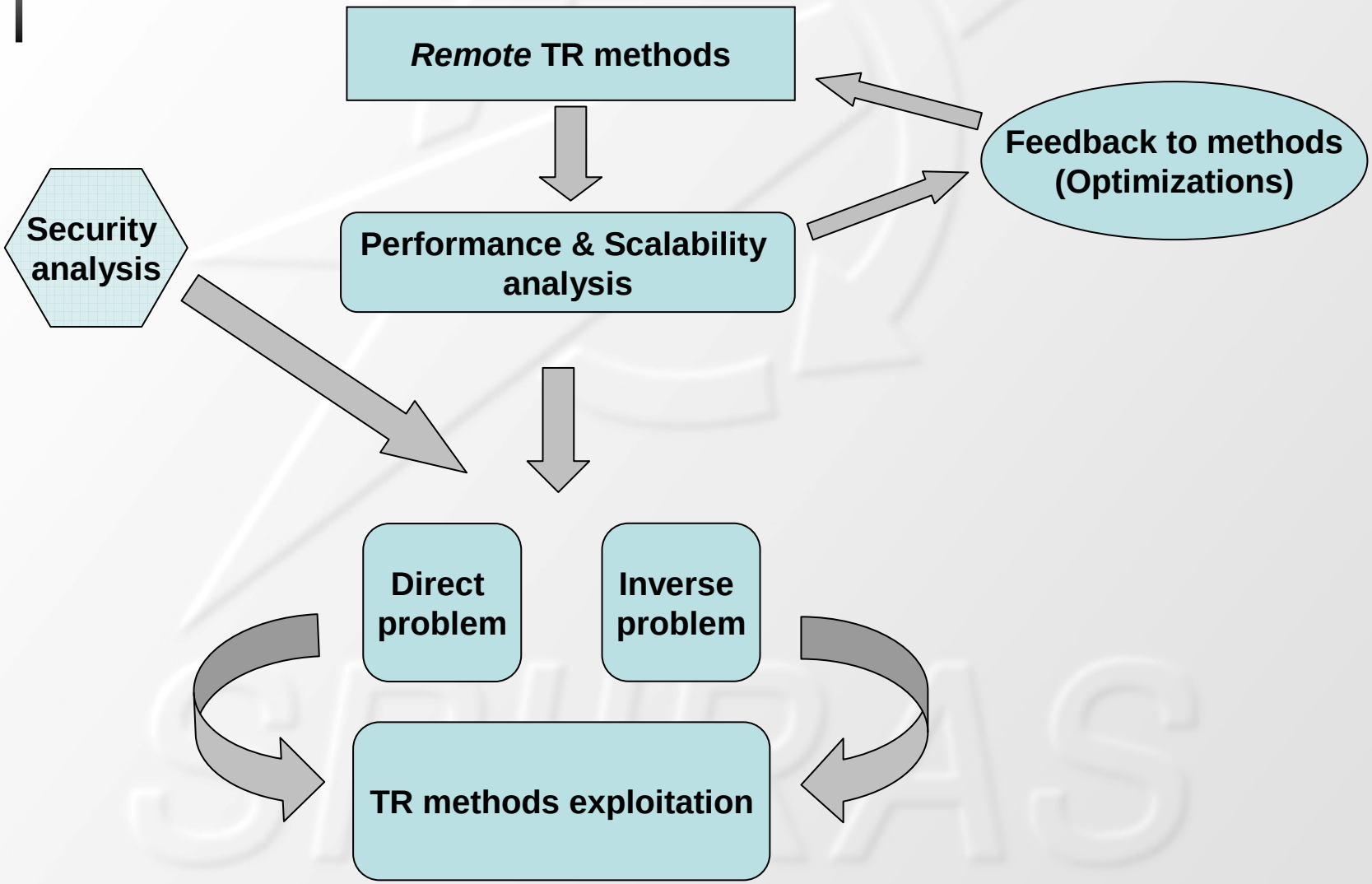
TR methods optimizing

- For each *Remote* TR method to learn
  - if the activities of the method (or a part of them at least) could be fulfilled *in advance*, before the client programs start
    - The positive answer would mean the required actions on the trusted entity could be accomplished one time only (*e.g. on the deployment phase of the mechanism*). Therefore these actions will not influence essentially upon the overall performance when client amount increasing
    - E.g. methods without replacement: *CFC, IC, BS*
  - If the activities of the method could be carried out for *multiple* clients *at once* (or for some groups of them at least)
    - The positive one would mean the performance of these actions don't depend on the client amount
    - E.g. methods that are not specified for each clients individually: *OR*, *BS partly*, (i.e. particularly the methods without remote attestation)

TR methods choosing

- The task of choosing concrete TR techniques and their combinations, depending on available server's resources amount
  - *Direct problem*. When the server's resources amount and maximum possible quantity of clients functioning simultaneously are fixed, we need to determine possible suitable combination of TR-techniques to apply and thereby to determine the security level, which could be provided.
    - This task comes to a discrete loading task (math extreme problem)
  - *Inverse problem*. Having some TR-techniques chosen we need to determine maximum amount of clients that the server is able to serve, guaranteeing the proper security.

Performance & Scalability Analysis workflow



Performance metrics definition

- Producing metrics determining resources consumption of each method
  - Central processor resources
    - Required time for the method
    - Rate – how many method copies could be performed concurrently under specific resources constraints
  - Memory amount required by the method
- Theoretically
  - By considering the most essential method's operations and estimation their complexity
- Empirically
  - On a test application to determine the difference between pure variant and protected one
  - Using tools prepared for performance assessment

TR methods performance determination (1/2)

- Combined approach
  - both theoretical and empirical features
- For the Remote methods being implemented in the project within SW prototypes
  - Pure empirical estimation is enough
- For the methods with no implementation
  - Theoretically: for each TR method to gather information characterizing all the most essential resources consuming operations being executed on the trusted server side
  - Empirically: model the trusted server side for the method by implementing these operations only and make corresponding empirical estimations (as in the first case)

TR methods performance determination (2/2)

- We would ask each partner in charge of each Remote TR method to provide us with that information
- In the first place we are interested in those operations that are fulfilled particularly for each client, so these are very critical for the scalability problem
- Two types of operations
  - Proper verification procedures checking client tags
  - Operation responsible for creation of replaceable components, which are sent to the client (for the methods *with replacement* only)
- Choose some applications to use as test ones

Security policies (1/3)

- Strategies of the trusted server's behavior in case of a lack of server's resources
- The goal is to define different possible *security policies* under witch the server could function
- Two main possibilities
  - Forced constraining of the amount of clients being served at the same time
  - Brief reduction of the granting security level of the mechanism

Security policies (3/3)

- Relatively short reduction of the granting security level of the mechanism
  - Equal decrease of the protection for all the clients
  - Protection reduction some group of clients
    - Clients having the best *security reputation* (e.g. no tampering before)
    - Different types of reputation (e.g. some information about the person/company using the client program)
    - To delegate the process of decision taking from the protection mechanism to the target application
  - Complex approach where
    - all clients are divided into groups (maybe with non-empty intersections)
    - for each group a specific policy is determined

Security policies (2/3)

- Forced constraining of the amount of clients being served at the same time
  - If the client program is able to function off-line (without server's support) some time period
    - To throw off the clients with the best *security reputation*
  - Otherwise and in the case when it is forbidden for a client to work without server's control
    - To throw off the less important clients for the program holder
      - Clients with trial/demo versions of the program and then unprivileged ones
      - Clients with the lowest *reputation*
      - To delegate the process of decision taking from the protection mechanism to the target application

Future work

- In-depth and detailed performance & scalability analysis
- Assessments of performance of TR methods, using existing toolkits
- Security estimations problem
- Determine possible TR methods combinations to apply
- To develop a toolkit that allows for a system designer to find an individual compromise between the required security and performance of the system, choosing certain TR techniques to deploy