Report of Task 3.2 break-out meeting

KUL – Dries, Thomas, Jan, Brecht
POLITO – Stefano, Paolo

Task objectives

- Use of light-weight HW to ensure software confidentiality and software integrity
- Light-weight HW:
  - TPM, SC, USB dongle

Discussions outline

- Advantages of using HW
- SW Confidentiality
- SW Integrity
- Practical solution

Advantages of using HW

- Controlled latency
  - Adv: possibly better time-based verification
- Delegated verification
  - Adv: scalability
  - But: not necessarily cheaper
- Identification
  - Adv: diversification; impossible identity theft
  - But: proxy attack still possible

Identification

- Issues
  - proxy attack (through back door)
  - identity theft (considered impossible to extract the secret key)
- Possible solutions
  - Proxy:
    - Limit possible nr of identifications
    - Use of controlled latency
  - Theft:
    - Confidential channel from server to HW

SW Confidentiality

- Hide original program
  - Software splitting
  - Code decryption on HW
    - But: dynamic analysis eventually reveals 'all' code
  - Hide control flow information
  - Data (critical variables)
- Hide monitor functionality
  - Examples:
    - Computation of invariants
    - Checksum algorithms

SW Integrity

- Confidentiality requirements usually imply integrity requirements
- Sometimes integrity is required without confidentiality being required.

How to transfer integrity from the trusted HW to the whole program.

Practical solution is ongoing research within track 3.2

Practical

- SW integrity verification based on invariants.
  - Tracing variables
  - Verification of invariants (using traces of variables)
- Server delegates parts of invariants verification to the HW

Practical (2)

- Extensions:
  - hide which variables are 'traced' (trace more, and filter in HW)
  - dynamically replace verification algorithm
  - use of probabilistic encryption (=> attacker does not know what the result means)
  - use of challenge-response system