Second year review
WP3 overview
HW/SW-based methods

Trento – October 17th, 2008
Goal

Investigate the combination of hardware- and software based software protection techniques in order to implement the remote entrusting principle
Participants

- KUL (WP leader)

Team:
- Bart PRENEEL
- Jan CAPPAERT
- Sebastian FAUST
- Thomas HERLEA
- Dries SCHELLEKENS
- Brecht WYSEUR
Participants

- KUL (WP leader)
- Gemalto

Team:
- Jean-Daniel AUSSEL
- Jerome D’ANNOVILLE
- Christian Cudonnec
Participants

- KUL (WP leader)
- Gemalto
- UNITN
  - Team:
    - Paolo TONELLA
    - Mariano CECCATO
    - Jasvir NAGRA
    - Milla DALLA PRED
    - Amitabh SAXENA
Participants

- KUL (WP leader)
- Gemalto
- UNITN
- POLITO

Team:
- Stefano DI CARLO
- Alberto SCIONTI
Participants

- KUL (WP leader)
- Gemalto
- UNITN
- POLITO
- SPIIRAS

Team:
- Igor KOTENKO
- Vasily DESNITSKY
Tasks

M0 M3 M6 M9 M12 M15 M18 M21 M24 M27 M30 M33 M36

T3.1  T3.2  T3.3  T3.4  T3.5  D3.2
Task 3.2
Hardware/Software Co-Obfuscation

Use of light-weight hardware to ensure software confidentiality and software integrity

- TPM, Smart card, …

Developments

- TPM assisted remote software entrusting (KUL)
- FPGA-based remote entrusting (POLITO)
- HW Barrier Slicing (UNITN)
TPM Assisted Remote Software Entrusting (KUL)

Trusted computing approach: remote attestation

- CRTM
- BIOS
- OS loader
- OS
- Application
- Network

TPM

Option ROMs

Hardware

Memory

New OS Component

root of trust in integrity measurement

trusted component

logging methods

measuring

reporting

storing

values
Disadvantages of timing based attestation techniques

- Constraints on verification function implementation
  - predictable execution time (interrupts, supervisor mode)
  - time-optimal
- Known hardware configuration $\rightarrow$ hardware replacement attack
- Network delays need to be incorporated $\rightarrow$ proxy attacks

Minimal trade-off: assist software attestation with TPM features.
TPM Assisted Remote Software Entrusting (KUL)
Enhanced solution: TPM tick stamping

\[
\begin{align*}
\text{Trusted platform} & : \quad c := \text{cksum}(TS_1,M) \\
& \quad h := \text{hash}(TS_2,P) \quad \text{M} \\
\text{Untrusted platform} & : \quad c := \text{cksum}(TS_1,M) \\
& \quad TS_1 := \text{Sign}_{\text{TPM}}(n||t_1) \\
& \quad TS_2 := \text{Sign}_{\text{TPM}}(c||t_2) \\
& \quad h := \text{hash}(TS_2,P) \quad \text{P}
\end{align*}
\]

\[t_2 - t_1 < \Delta t_{\text{expected}}\]
TPM Assisted Remote Software Entrusting (KUL)

Extensions: assistance for trusted OS loader

- Include HW specifications (CPUID) in Tag
- Simulate verification function at boot-time

Publication

FPGA-based remote entrusting
(POLITO)

System Architecture

- CLIENT APPLICATION uses available services exported by the DRIVER
- DRIVER manages communication between the application server and the authentication hardware
- AUTHENTICATION MONITOR manages the application code hashing and encrypting operations
FPGA-based remote entrusting (POLITO)

1. At startup of the client application, a session key is established, using a key agreement protocol between the application server and the client machine. Optionally, the session key can be updated during the execution of the program.

2. The session key is used to compute a signature of the client application.

3. The server periodically sends to the hardware monitor an Authentication Request, and waits for the computed signature.

4. The client receives the requests (on a socket interface implemented in the driver module) and forwards it to the hardware monitor.

5. The hardware monitor computes the hash of the memory pages’ content related to the client application (code segment) directly accessing the computer’s memory and without relying on any system call. The only used information is the name of the target application used to determine the position of the application in memory.

6. The hardware module computes the signature for the considered memory pages using the session key, and sends it to the server via the driver’s socket.

7. The server compares the two signatures and determines whether it can already deliver the service to the client or not.
FPGA-based remote entrusting (POLITO)

Application Server

Client

FPGA

session key agreement

request

$\text{sign}_{sk}(\text{mem})$

service

hash

mem
Distributed Architecture
(UNITN)

Un-trusted host

Network

Program P

Card Reader

Virtual secure channel

Trusted host
T3.2 Hardware/Software Co-Obfuscation

Program Transformation
(UNITN)

Un-trusted host:
- $X \in s_{\text{unsafe}}$
- X uses are removed from the program
- They are replaced by a query to get the actual value from the card
- X defs are replaced by synchronization statements

Smart card:
- The barrier-slice is run
- The slice is fed with any input coming from the host
- Validity of the host is evaluated
- X values are provided as required
- Synchronization with the host
Empirical results
(UNITN)

<table>
<thead>
<tr>
<th>Original client</th>
<th>Barrier slice</th>
</tr>
</thead>
<tbody>
<tr>
<td>858</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td>14%</td>
</tr>
</tbody>
</table>

- **Memory**
- **Threads**
- **Network**
Empirical results
(UNITN)

• Barrier slicing is used to separate the security sensitive part of the application

• Both centralized and distributed architectures are able to verify the client healthy execution

• The distributed architecture has better scalability
  15% memory
  25% threads
  8% network

• The slice is small and can fit in a smart card (14% of the application)

Mariano Ceccato, Jasvir Nagra, Paolo Tonella, “Distributed Trust Verification to Increase Application Performance”, In 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing, 2008
Task 3.3
T3.3 – Encrypted Code Execution (KUL - GEMALTO)

Computing with Encrypted Data

- State of the art study (Goldwasser-micali, Paillier cryptosystems, Boneh)


- Relation with White-Box Remote Program Execution (T2.4)

Smart Dongle
Secure with hardware
(GEMALTO)

• Scope

Study the opportunity to use a new hardware as a candidate platform for the project/task

• Platform:

  USB Dongle: Smartcard + flash memory

• Purpose:

  - Use the platform in the context of the T3.3
  
  - Host the monitor locally in the USB Dongle
Secure with hardware
(GEMALTO)

- Evolution of the USB Dongle: no more hub
T3.3 Encrypted code execution

Secure with hardware
(GEMALTO)

- USB Dongle: Smart Card + Flash memory
Secure with hardware
(GEMALTO)

- USB Dongle main features:
  - No installation, Zero footprint
  - Smartcard
  - Levels of trust
    - CD-ROM partition: no persistent tampering
    - Secure channel
    - Thumbprint mechanism to secure the application
Untrusted computer

Service:
- Data Access
- Cryptography

Property Analyzer

Application maintenance

Trusted Application
- Keys
- Memory
- Thumbprints

Monitor

Server
- Log service
- Releases Mgt

Provide or shutdown service

Report, update properties

Smartcard
Secure with hardware (GEMALTO)

• Smart card entrusting of terminal applications

T3.3 Encrypted code execution

Smart Card

Untrusted Computer

Properties Analyzer

Report Properties Update Properties

Trusted Application

Monitor

Keys Memory thumbprints

Provide or shutdown service

Service: Data Access Cryptography
Secure with hardware
(GEMALTO)

- Attacker may switch code segment pointer to fool the thumbprint mechanism
- No ideal white box-cryptography: Master keys could theoretically always be retrieved
- Session keys are stored in RAM during processing
Tasks 3.4
Observable Cryptography (KUL)

- Traditional Provable Security
- Cryptographic primitives behave as black-boxes
- Gap with real world: physical devices leak information (side-channel attacks)
Micali / Reyzin Model

Promising model
Based on reduction proofs
- Micali / Reyzin studied basic theoretic constructions
- PO OWF $\Rightarrow$ Unpredictable Generator
- Disadvantage: inefficient constructions, not used in practice

Our work
- Analysis of constructions that are used in practice (and that have security proofs)
  - Develop security notion
  - Think which assumptions we need to achieve these security notions
  - Prove (within model)
Micali / Reyzin Model

Real-world model

Secure basic primitive

Study of practical constructions

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Assumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA-CPA (which is IND-CPA)</td>
<td>RSA one-way assumption</td>
</tr>
<tr>
<td>RSA-OAEP (which is IND-CCA)</td>
<td>Unforgeability assumption</td>
</tr>
<tr>
<td>RSA-FDH</td>
<td>Strong unforgeability assumption</td>
</tr>
</tbody>
</table>
Micali / Reyzin Model

Good news
- We have proved the security of these constructions in the PO model

Bad news
- For each step, extra assumptions need to be introduced
- Assumptions are tailored for particular problem and do not reflect practice
- We want something like one primitive that allows to build lots of constructions

Further work
- Change model
- Develop new schemes (though not likely; will face similar problems)
Ishai Model
(Private Circuits I)

Model

- Boolean circuit implementation
- Adversary can adaptively probe t wires
- t-security – whatever A can do, can be done without side-channel information → no additional information obtained

Provably Secure Implementations presented

- Based on secret sharing
- Result: circuit with $O(nt^2)$ gates (n is original number of gates)

Conclusion: probing model is unrealistic. In practice, a more serious threat is power analysis, where the adversary obtains the joint power consumption of several wires for one measurement and thus a single measurement can depend on all shares ==> no provable security can be achieved

Details on Micali / Reyzin and Ishai model research will be delivered in extensive report in Y3.
Task 3.5
• Generally, the task investigates the possibilities to combine the different techniques developed and analyzed in WP2 and WP3 to one overall approach.

• According to the plan this task is on its initial phase
T3.5 – scalability and performance

(SPIIRAS)

• Two crucial objectives facilitating remote entrusting principle:
  • Specific security level
  • Overall system performance and scalability

• The aim is to combine these two objectives to obtain some specific tradeoff on real applications
T3.5 – scalability and performance

(SPIIRAS)

- System designer should have a chance to choose a bundle of TR (Tamper Resistance) techniques representing the “security/performance” tradeoff for the concrete application.

<table>
<thead>
<tr>
<th>Application</th>
<th>System designer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application characteristic</td>
<td>take into account</td>
</tr>
<tr>
<td>Hardware in use</td>
<td>SW, SW/HW based TR techniques and methods</td>
</tr>
<tr>
<td>Techniques characteristic</td>
<td>take into account</td>
</tr>
</tbody>
</table>
T3.5 – scalability and performance

(SPIIRAS)

- Specific security level
  - achieved by combination of SW and SW/HW based TR techniques and dynamic replacement principle

- Overall performance consists of
  - client program performance
  - Trusted Server performance (may interacts with a great deal of clients)
  - Medium network performance
Activities influencing upon the overall performance to the utmost:

- TR techniques demanding complex computations (both on client and server). \textit{E.g.: barrier slicing, data encoding/decoding, etc.}

- network connection bandwidth. \textit{E.g.: barrier slicing could require a lot of data passes, and so great network resources}

- HW devices work, which could slow down the client performance. \textit{E.g. smartcards, etc.}
T3.5 – scalability and performance

(SPIIRAS)

Original Re-Trust objective:

(*) security determines replacement

(Replacement period is determined from the security quality)

For T3.5 we suggest two objectives:

(**) security & replacement determine Hardware

(To determine minimal necessary Hardware capable of meeting the given security and replacement qualities)

(***) Hardware determines security & replacement

(Having some specific Hardware to determine the security level which could be reasonable to provide)
Performance and Network
(SPIIRAS)

- For the (**) and (***) records from the previous slide

- An important dependency connected with the performance:
  - If the amount of the TR techniques in use and theirs robustness increase then the value of acceptable replacement period also increases.

- So, it means that security and replacement can increase, however the network hardware could remain the same
Performance and Trusted Server (SPIIRAS)

- Actually it is one of the most important issues for RE-TRUST mechanism to be applied in practice.

- So, for the case when a huge number of unique clients are connected with the same server the system designer should choose TR techniques free of any serious server based computations only.

- An individual case is the one where the server certainly has constrained amount of clients (e.g.: application working within the limits of an enterprise network).
Scalability and performance

The task investigates the possibilities to combine the different TR techniques developed and analyzed in WP2 and WP3 to one overall approach.

- Two objectives:
  - Specific security level achievement
  - Overall system performance and scalability

- System designer should have a chance to choose a bundle of TR techniques representing the “security/performance” tradeoff for the concrete real application
Scalability and performance

Specific security level

- achieved by combination of SW and SW/HW based TR techniques and dynamic replacement principle

Overall performance consists of

- Client program performance
- Trusted Server performance
- Medium network performance

Activities influencing upon the overall performance to the utmost:

- TR techniques demanding complex computations (both on client and server). *E.g.: barrier slicing, data encoding/decoding, etc.*
- network connection bandwidth. *E.g.: barrier slicing could require a lot of data passes, and so great network resources*
- HW devices work, which could slow down the client performance. *E.g. smartcards, etc.*

To determine minimal necessary HW/OS capable of meeting the given security and replacement qualities

Two objectives:

*Having some specific HW/OS to determine the security level which could be reasonable to provide*
Future activities

- Implementation of techniques on the Trusted Platform (smart card, and FPGA) (T3.2)
- Development Smart Dongle and Techniques for Computing with Encrypted Data (T3.3)
- Improved model for Physically Observable Cryptography (T3.4)
- Analysis of the different approaches and possibilities of their combination (T3.5)