Session 5
Cloud Security and Privacy

Presenters: Christof Fetzer, Marco Vieira

Rapporteur: Antonio Casimiro

SCONE – Secure Containers

- Presented by Christof Fetzer
- From a service provider perspective, a fundamental requirement is to protect data confidentiality and integrity
  - The cloud is supposed to provide availability
- Threat model
  - Cloud and development machines cannot be trusted
- The presented SCONE platform simplifies running applications in SGX enclaves
  - Dealing with attestation and configuration
  - Without the need to modify applications (several tool chains for building obtaining SCONE-enabled applications)

General approach

- Applications are partitioned into μ-services
- A μ-service is deployed in a container
- The container can be made (if needed) secure by running in a SGX enclave
  - Outside the processor data is always encrypted
- The SGX enclave is executed in a SGX host

Partitioning

- The trend is to use μ-services for several reasons
  - E.g. modularity, confinement, etc
- Should applications be partitioned into μ-services?

- Example considering nginx:
  - It is not sufficient to protect TLS, www files also need to be encrypted, modifications must be detected
  - Conclusion is that it is easier to put entire application in container instead of partitioning

Protecting against software bugs

- Main approaches:
  - Bounds checker
  - Isolation of μ-services
  - APIs with limited access
- When applications are partitioned into several μ-services, it is not sufficient to protect (in a SGX enclave) the μ-services containing secrets
  - Other μ-services with credentials may contain vulnerabilities and may be compromised, ultimately being used to access protected data
  - It is also not sufficient to protect μ-services containing credentials
  - Conclusion: run all μ-services in enclaves and harden external APIs

Additional issues

- Several toolchains for constructing protected applications
- Containers are lightweight compared to VMs
- Several SCONE curated images available
- The “only” thing that is needed is Docker Swarm

Discussion

- Question and discussion about the need for protecting all μ-services, entire application, vis-à-vis protecting only what is essential
- Question about data owners trusting μ-services: OK if they can trust (by attestation) that the μ-service is what they expect
- Performance penalty somehow mitigated due to data being held in a page cache (OK if all fits in)

ATMOSPHERE – Resilient Cloud Services

- Presented by Marco Vieira

- Definition of trust
  - Dynamic property that changes over time
  - Trust: relation between two parties
  - Trust from different perspectives (security, privacy, coherence, isolation, stability, fairness, transparency, dependability)

ATMOSPHERE project

- Nice acronym ☺
- Started recently (EU/Brazil H2020)
- Project involves trustworthiness assessment (& monitoring framework)
- One objective is to define a trustworthiness life-cycle
- Three main work areas:
  - Hybrid and federated platform for trustworthiness
  - Trustworthy data management services
    - Preserve privacy, based on enclaves – related with previous presentation
  - Data processing services
    - Data analytic techniques for data processing
    - Dealing with privacy requirements (paying special attention to how to handle compositions, avoiding information to be extracted from relations)

Trustworthiness Framework

- The trustworthiness evaluation framework is orthogonal to the three main building blocks and deals with properties and metrics

- Properties
  - Table with several properties, and each property with several attributes
- Metrics
  - Each property/attribute may be quantified by different metrics
  - An algebra (or possible several different algebras) to obtain final scores for properties, based on evidences for basic attributes, is(are) required
- Properties can be evaluated for each layer (platform, data management, data processing)

Multidimensional approach

- Trustworthiness can be evaluated in several ways
  - Overall trustworthiness
  - Considering a specific perspective
  - Etc.
- And it can be evaluated in:
  - Design-time: through testing, static analysis, modelling, etc
  - Run-time: through continuous monitoring, MAPE-K cycle

Examples on possible metrics

- Trustworthiness from a performance perspective
  - A relevant metrics could be the time needed to execute a service: provides indication on response time and availability
- Trustworthiness from a privacy perspective
  - Big question mark…

- Some challenges:
  - Define attributes
  - Define metrics
  - Define algebras to handle metrics for each attribute (trustworthiness models)
  - Etc.

Discussion

- Question about how is data collected in run-time, for monitoring
- Question about adaptation and trustworthiness changes in run-time and goes below some threshold

- How to evaluate the validity of the evaluation method?