Developing secure applications with metrics in mind

Thomas Heyman, Christophe Huygens and Wouter Joosen

DistriNet, dept. of Computer Science
K.U.Leuven - Belgium

thomas.heyman@cs.kuleuven.be
How secure is my application?

• Our focus is on application security
  • White box (vs. black box)
• “How secure is my application?”
  = Do security requirements still hold?

☑️ Measuring if security mechanisms work as intended
☐ Not measuring, e.g., #blocked intrusion attempts
Measuring (application-level) security

1. Need a structured repository of metrics
   - Collect time-tested metrics

2. Need a framework for measuring
   2.1. Facilitating selection at development time
   2.2. Aggregate/interpret according to security objectives
Outline

• Problem: how secure is my application?
• Our solution
  • Associating metrics to security patterns [1]
  • Instantiate metrics through security patterns [2.1]
  • Interpret measurements during production (or development) [2.2]
• Conclusion and future work
Associating metrics to security patterns [1]

- Security patterns:
  - Package domain-independent knowledge and expertise
  - Reusable!
- Possible to attach security metrics to patterns [MetriCon1]
  - Ecosystem vs. core
- Use pattern selection to piggy-back metrics in the application

[MetriCon1] Software Security Patterns and Risk, T. Heyman and C. Huygens
Integrating metrics in the development cycle [2]
Metrics selection [2.1]

- Security requirements (domain specific) are assigned security objectives (domain independent)
- Select coherent set of security patterns to realise objectives (e.g., using [Yskout])
- Implement associated metrics

Metrics selection - illustration

Security objectives (and sub-objectives)
- Auditing
- Non-repudiation
- Principal authentication

Security patterns (and metrics)
- Audit Interceptor
- Secure Logger
- Authentication Enforcer

AND
OR

Metrics selection
Interpreting measurements [2.2]

- Use associated security pattern to aggregate measurements
- Each pattern is instantiated for a certain security objective
  - E.g. Audit Interceptor provides Auditing
- Patterns might depend on other patterns/objectives
  - E.g. Audit Interceptor depends on Secure Logger
- Combine measurements through resulting AND-OR-graph
Interpreting measurements - illustration

Security objectives (and sub-objectives)

Accountability

Auditing

Non-repudiation

Principal authentication

Audit Interceptor

Secure Logger

Authentication Enforcer

Measurement interpretation

Security patterns (and metrics)

AND

OR
Conclusion

• A gap exists between:
  • high-level security requirements (i.e., what do stakeholders want)
  • production-level measurements (i.e., what is happening)

• Security patterns help to bridge this gap
  • Facilitates metric selection and instantiation
  • Enables aggregation of measurements to high-level indicators
Ongoing and future work

- Further validation: developing a PoC ATM system
- Perform sensitivity analysis on dependency graph
  - Identify “key indicators”, weak points
- Seamlessly integrate metrics in code (through AOP)
  - Automation?
Thank you!

Questions or remarks?

thomas.heyman@cs.kuleuven.be