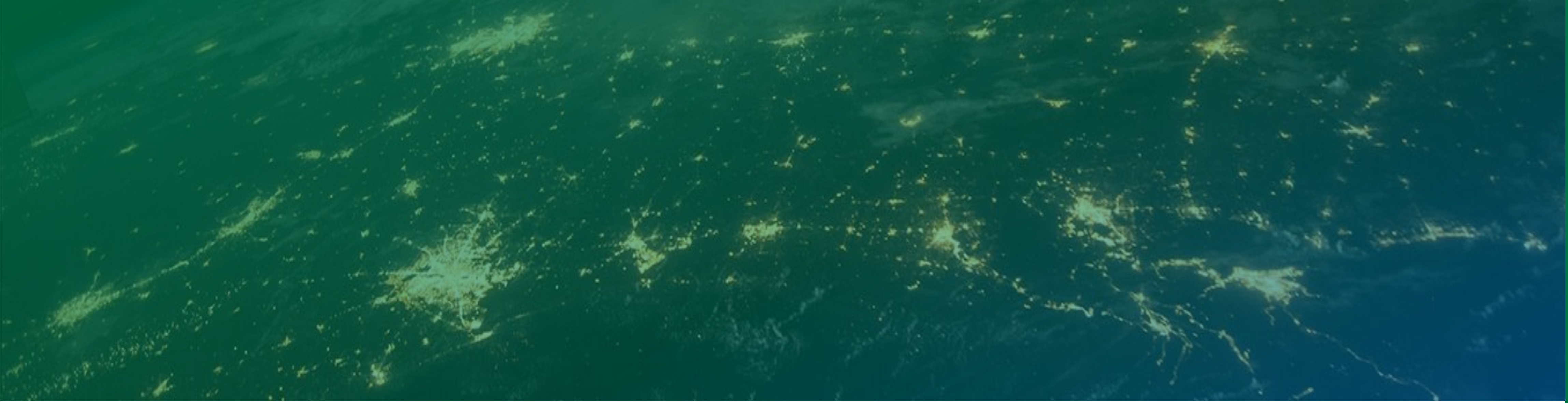


Yasamin Mahmoodi[1], Christoph Groß[1], Sebastian_Reiter[1], Alexander Viehl[1], Oliver Bringmann[2]

[1] FZI Research Center for Information Technology
[2] University of Tuebingen

Presenter: Yasamin Mahmoodi, Email: Mahmoodi@fzi.de

The Fifth International Conference on Cyber-Technologies and Cyber-Systems CYBER 202

SECURITY REQUIREMENT MODELING FOR A SECURE ENERGY TRADING PLATFORM

Bio

Yasamin Mahmoodi

Institute: FZI (Forschungzentrum Informatik)

Department (ISPE) Intelligent Systems and Production Engineering

University: University of Tübingen

Country: Germany

Position: Doctoral candidate in embedded system's security

Email: mahmoodi@fzi.de

Research area

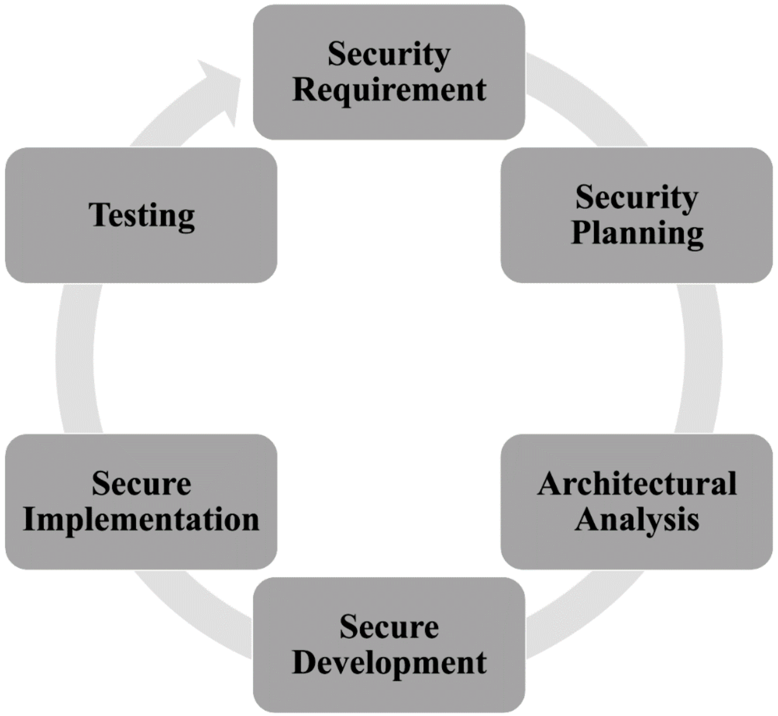
- Security of Internet of Things
- Security analysis of embedded systems
- Architectural analysis
- Static and dynamic analysis
- Security modeling for embedded systems
- Virtual prototyping
- Penetration testing

Motivation

- Connected Internet of Thing (IoT)
  - Offer numerous advantages
  - Also bring up new security challenges and threats
    - financial data
    - medical data
    - passwords
  - Safety and security consideration required
- Smart energy market
  - Decentralized small-scale marketenergy
  - Photovoltaic system on the roof for households
    - produce their own energy
    - sell the rest to their neighbors
  - Utilization of the IoT paradigm to create an automated local energy trading market

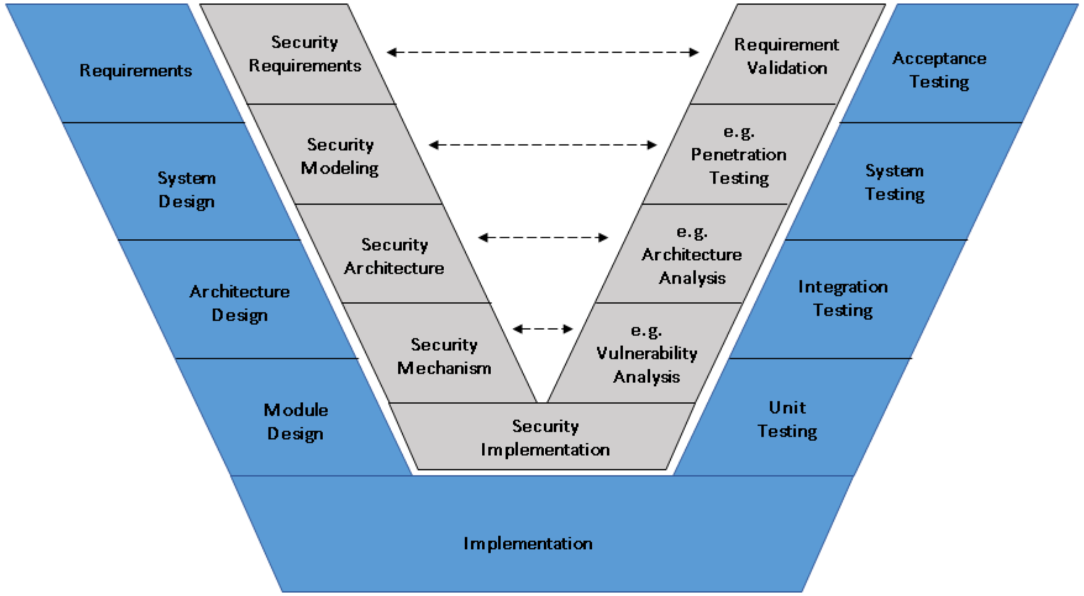
Secure System Development Life Cycle (SSDLC)

- SSDLC defines tasks such as:
  - The definition of security requirements
  - Assessing their risks
  - The planning of the security architecture
  - The actual design and implementation
  - Task regarding testing and security assessment



Security-Based V-Model

- Traditional V-model
  - Testing phase associated with each development phase
  - The blue diagram represents the traditional V-model
    - system development on the right side
    - testing phases on the left side



- mapping the SSDLC to the V-model design flow
  - the iterative nature of the SSDLC should be applied to the traditional V-model
  - Resulting in a repeated adjustment and refinement of the system

Security Requirements of Embedded Systems

- The requirement specification
  - the entry point of a system development process
  - the standard between stakeholder's requirement validation, development and testing
- It specifies goals, functions and constrains of the system

- Standard categories for security requirements:
  - Confidentiality
  - Integrity
  - Availability

Proposed approach

- A three-level requirement modeling
- Sequential abstraction layers principle of the V-model
- Track of the security requirements
- Starts with general security considerations
- Continues to explain in more detail
- Applied to an Unified Modeling Language (UML)

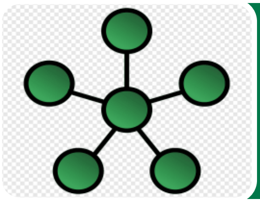
Level 1	General information about security needs
Level 2	Classifying protection goals in three general category of confidentiality, integrity and availability
Level 3	Tagging protection goals with stereotypes of security profile

Proposed approach; Level 1


Important parts of the system


Potential attackers


Potential entry points for attackers


Network topology of the system

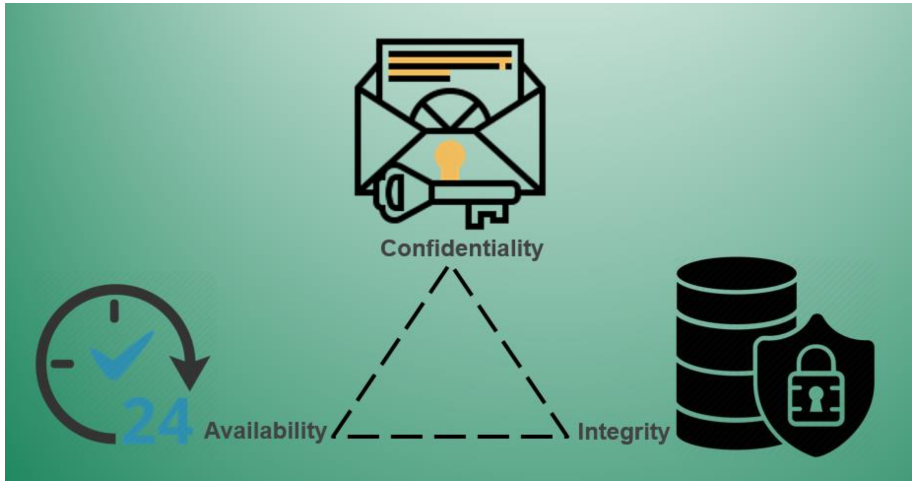

Security mechanisms


Security of hardware, software and technologies


Software updates

Proposed approach; Level 2

- Protection goal categorizing
- Based on CIA triad
  - Confidentiality: ensures that access to the critical data is available only for authorized users.
  - • Integrity: assures the correctness and completeness of the data over its entire life cycle.
  - • Availability: makes sure that data and services are available for authorized users.



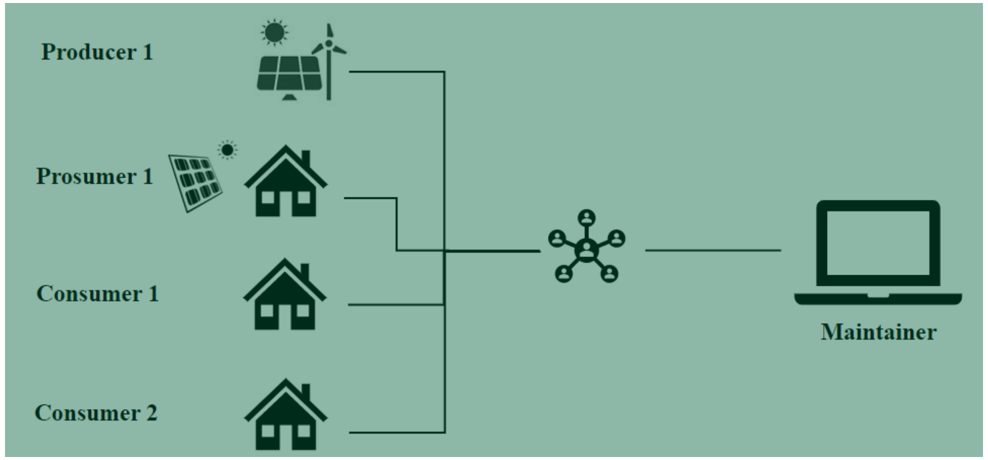
Proposed approach; Level 3

- Based on proposed security profile
- Documentation and threat modeling
- Protection goal categories:
- Confidentiality
  - Protection goal, data confidentiality (PG. C)
- Integrity
  - Protection goal, data modify (PG. M)
  - Protection goal, data add (PG. A)
  - Protection goal, data delete (PG. D)
- Availability
  - Protection goal, service availability (PG. Ava)

Stereotype	Base class	Tag
AS.DataRead	Property	Effort
AS.DataModify	Property	Effort
AS.DataDelete	Property	Effort
AS.DataAdd	Property	Effort
AS.ServiceUnavailable	Operation	Effort
PG.DataConfidential	Property	Severity
PG.DataModify	Property	Severity
PG.DataDelete	Property	Severity
PG.DataAdd	Property	Severity
PG.ServiceAvailability	Operation	ResTime
Doc.EncryptionProtocol	Class	String
Doc.SecurityProtocol	Class	String
Doc.SoftwareVersion	Class	String
Doc.Authentication	Operation	
Doc.Authorization	Operation	
DP.DataFlowElement	State	

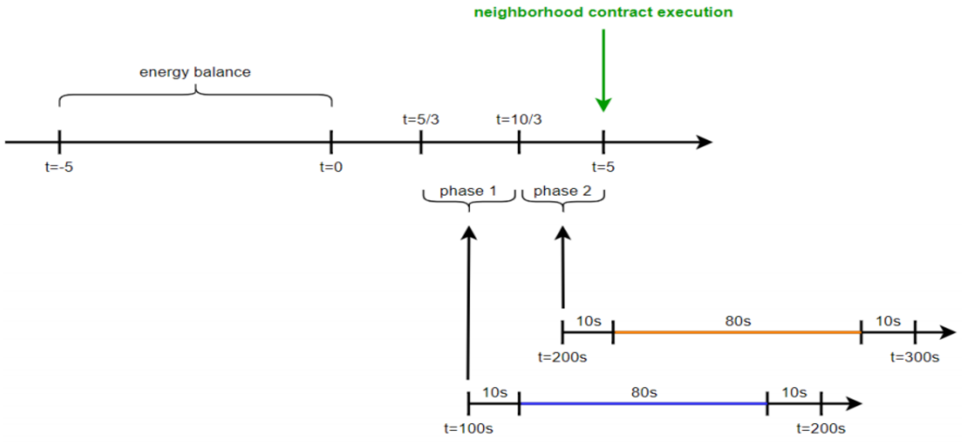
Case Study: enerDAG

- Energy Directed Acyclic Graph
- A local energy trading platform offered
- A platform for households to trade energy with their neighbors
- Highly distributed computing system
- Smart contracts and majority voting
- Nodes have positive or negative energy balance
  - Consumer : negative energy balance
  - Producer: positive energy balance
  - Procumer: positive or negative energy balance



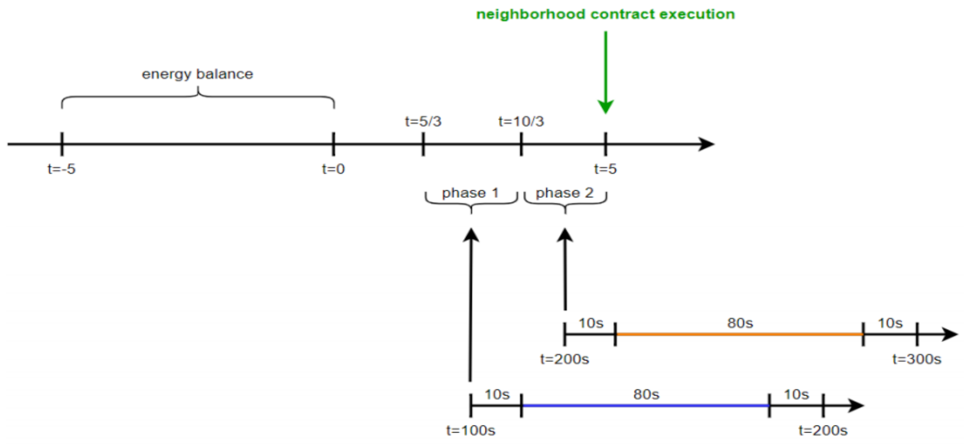
Case Study: enerDAG

- Five-minutes intervals
- Phase 1:
  - Searching for result of the previous market execution
  - Sending energy balance
  - Sending selling or buying price
  - Creating bid with a predefine structure
  - Encrypting the bid with private key of node
  - Encrypting the message with public key of neighborhood
  - Sending the transaction to the tangle



Case Study: enerDAG

- Phase 2:
  - Sending the private key of the node
  - Receiving message from other nodes
  - Decrypting the messages
  - Calculating the contracts
  - Sending the results t other nodes



enerDAG daemon

- Is installed on each node
- Establishing a database connection
- Running the main loop:

contractEngine()	connectionEngine()
• Runs every minute • Searches for contract to execute • Executes contract with contactExecuter() function • Sends the results to the node	• starts a server • listens on a port for incoming messages via the handleIncomingEvent() function. • Handles the incoming transactions

Security requirement analysis of enerDAG; Level 1

- Exemplary security demands of enerDAG are:

  - Secure energy trading
  - Transparent transactions
  - Anonymity of the participants
  - Non repudiation
  - Secrecy of consumed or produced energy
  - Secrecy of the offered price
  - Unauthorized user should not be able to participate in the market.
  - Authorized users should not be able to cheat.
  - A potential attacker : unauthorized or an authorized user

Security requirement analysis of enerDAG; Level 2

- Categorizing security requirements based on CIA triad

Confidentiality

- Energy balances of the participants
- Offered prices of participants
- The bids offered by participants
- Private key of the household nodes.
- Public key of the neighborhood
- The transactions
- The seed sent by the maintainer to the household nodes
- List of neighbors

Security requirement analysis of enerDAG; Level 2

- Categorizing security requirements based on CIA triad

Integrity

- Energy balances of the participants
- Offered prices of participants
- The bids offered by participants
- Contract execution
- Majority voting
- The tips (least two older transactions in tangle data structure)
- List of neighbors

Availability

- The server listening for new transaction
- Transaction handling
- Contract execution

Security requirement analysis of enerDAG; Level 3 Neighborhood node

- contractExecutor() as a service should be available
  - Asset: PG.Ava
  - Severity: Medium
  - Offered Security mechanisms: security policy (restricting sending message), IDS, firewall.
- Contract folder
  - Asset: PG.M, PG.A, PG.D
  - Severity: Low
  - Offered security mechanisms: Verifying integrity of the data using HMAC (Hash Message Authentication Code), AAA (Authorization, Authentication, Accounting) and to prevent hackers to be able to modify contract folder
- Majority voting
  - Asset: PG.M, PG.A
  - Severity: Medium
  - Offered security mechanisms: Encryption, hashing, verifying integrity of the data using HMAC, AAA

Security requirement analysis of enerDAG; Level 3 Neighborhood node

- Minimum selling/ maximum buying price in database
  - Asset: PG.C, PG.M, PG.A, PG.D
  - Severity: Medium
  - Offered security mechanisms: proper separation of database, encryption, hashing, verifying integrity of the data using HMAC, AAA
- Validation key
  - Asset: PG.C, PG.M, PG.A, PG.D
  - Severity: High
  - Offered security mechanisms: Proper separation of Database, encryption, hashing, verifying integrity of the data using HMAC, Key management AAA
- Bid
  - Asset: PG.C, PG.M, PG.A
  - Severity: Medium
  - Offered security mechanisms: Encryption, hashing verifying integrity of the data using HMAC, AAA

Security requirement analysis of enerDAG; Level 3 Maintainer node

- Neighborhood list
  - Asset: PG.M, PG.A
  - Severity: Medium
  - Offered security mechanisms: Hashing, verifying integrity of the data using HMAC, AAA
- Validation seed
  - Asset: PG.C, PG.A, PG.M
  - Severity: High
  - Offered security mechanisms: Encryption, hashing, verifying integrity of the data using HMAC, AAA, key management
- Neighborhood cryptography
  - Asset: PG.C, PG.M, PG.A
  - Severity: High
  - Offered security mechanisms: Encryption, hashing, verifying integrity of the data using HMAC, AAA and to prevent hackers to be able to modify contract folder

Security requirement analysis of enerDAG; Level 3 Maintainer node

- Contract address
  - Asset: PG.C, PG.M, PG.A
  - Severity: Medium
  - Offered security mechanisms: Encryption, hashing, verifying integrity of the data using HMAC, AAA

Conclusion

- Three abstraction levels for security requirement modeling
  - First level: general security issues
  - Second level: categorizing security goals base on CIA triad
  - Third level: detail classification of protection goals based on proposed security profile
- Applying proposed approach on a use case
  - enerDAG, a platform for smart energy trading


Thank you