



ARMOUR

Model-Based Security Testing for IoT Labelling & Certification

Elizabeta Fournaret (Smartesting)

Panos Karkazis (SYNELIXIS)

www.armor-project.eu

ReCRED, Clustering Workshop, Athens, Greece

2018, January 31st



Contribution for IoT Labelling and Certification

1. How to make the testing part of the labelling and certification process cheaper ?

- Approach **Easy to use** by certification bodies and **extensible**
- Build on reusable, configurable security test patterns and automated test generation
- The certification scheme comes with the test patterns to be used

2. How to ensure the quality and reproducibility of the assessment?

- The **security test patterns** should be agreed by the certification authorities
- Test automation ensure the replicability of the results

3. How to deal with change?

- Using the **automated testing** for continuous monitoring and testing at running stage to keep the certificate alive

ARMOUR in a nutshell

- Duration** 24 months (from Feb 2016 to Jan 2018)
- EU funding** 2 Millions €
- Consortium** 8 partners including 5 SMEs, 1 university and 2 research centres



Challenges of IoT Security & Trust

Business Logic Vulnerabilities

Uncertainty of the expected behaviour of IoT systems



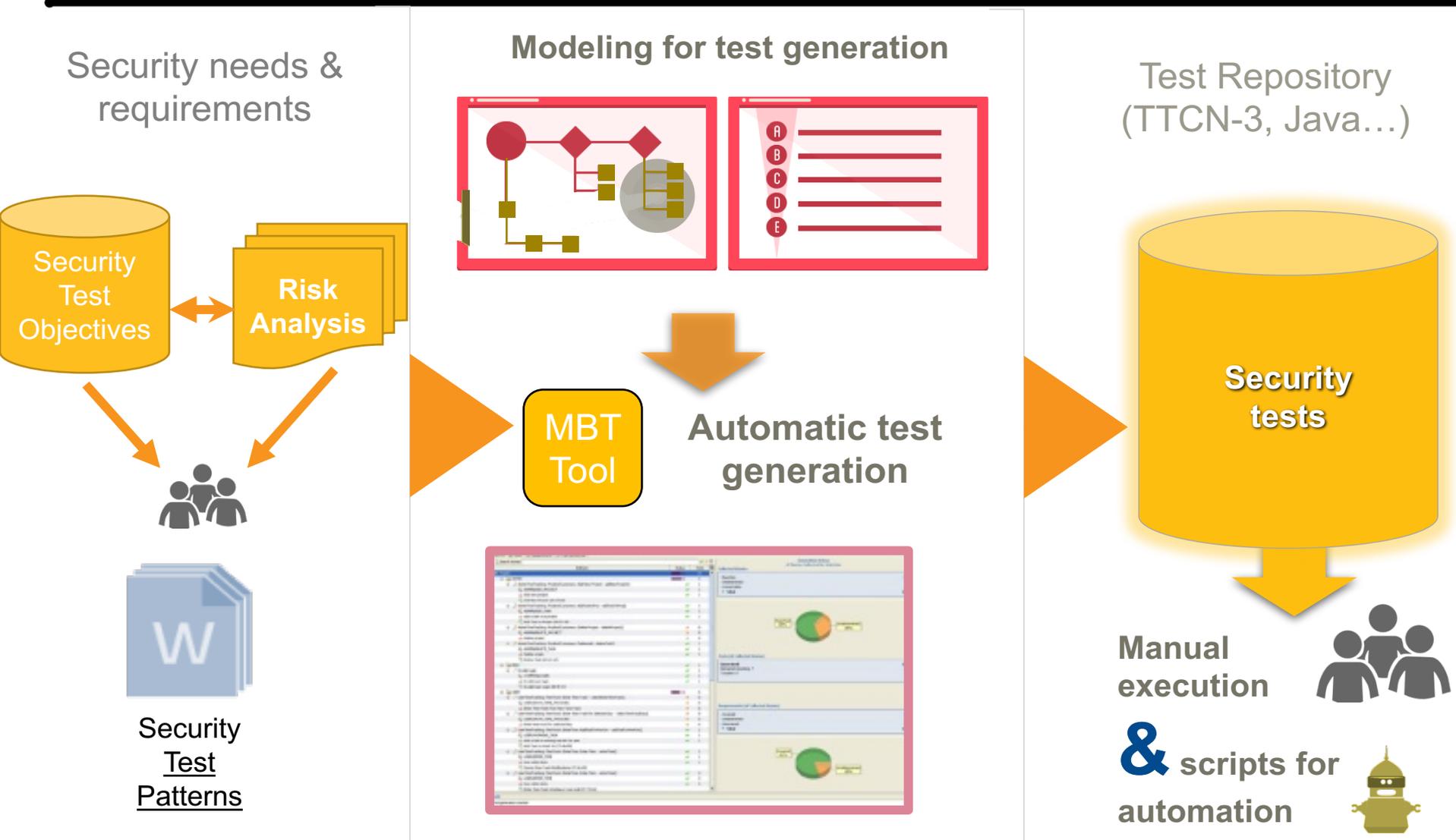
Large-scale dimension, heterogeneity, compositionality and dynamic configuration of IoT systems

Ensuring End-to-End Security & Privacy Testing

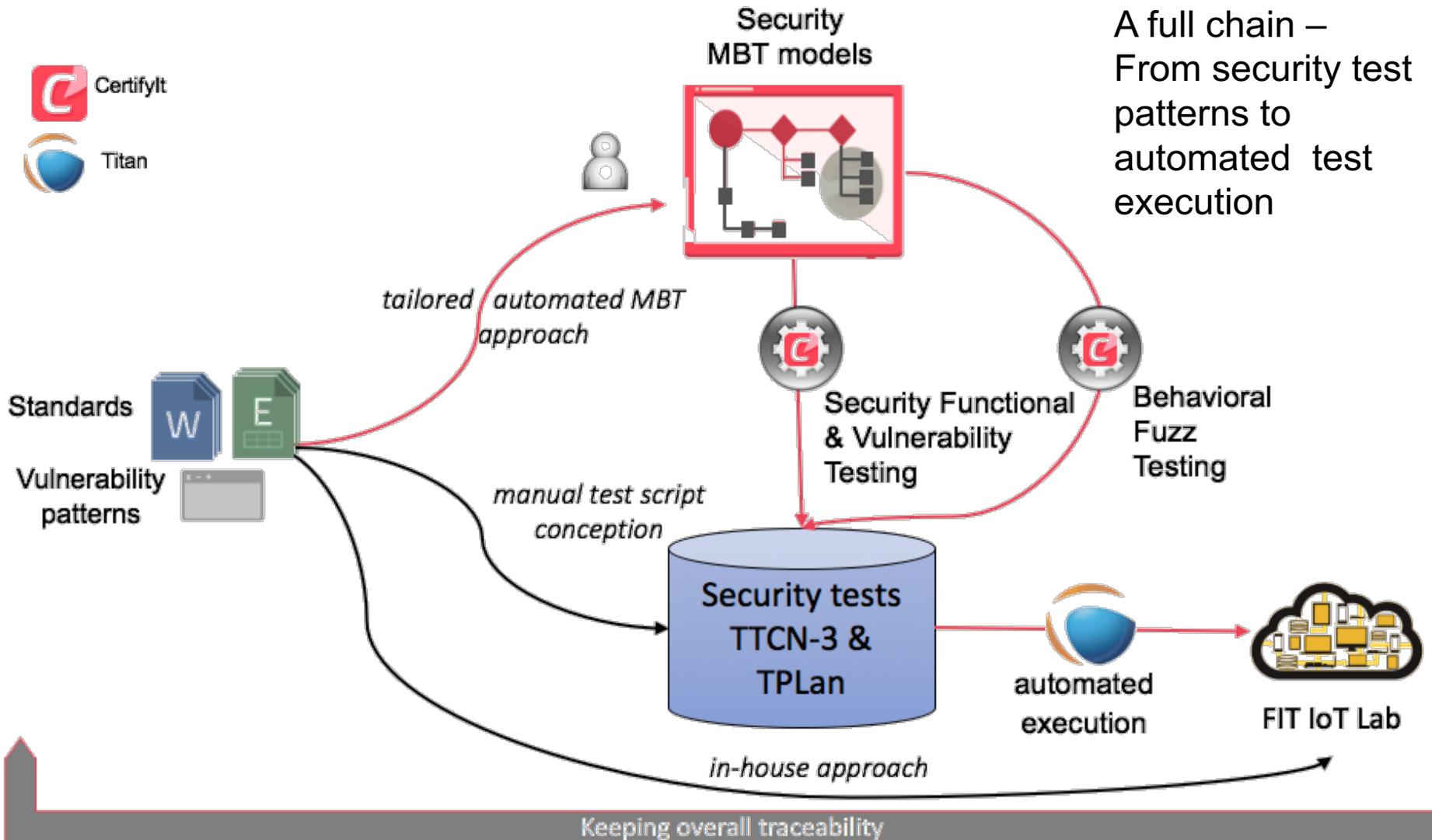
MBST Process



ARMOUR



ARMOUR Testing Framework



A full chain –
From security test
patterns to
automated test
execution

MBST Approach in 5 steps



ARMOUR

Test Pattern ID	TP_ID6
Test Pattern name	Run unauthorized software
Context	Kind of test pattern: design and test data Security approach(es): detection and prevention
Problem/Goal	The pattern addresses how to check that an entity should not run an unauthorized software if it does not pass a prior integrity verification process.
Solution	Test procedure template 1. Consider that an entity has downloaded unauthorized software



Test purposes

Test Purpose definition

Tags

for_each item \$response from INSTALLED_FIRMWARE or FAILED_SIGNATURE or NO_FIRMWARE_TO_INSTALL,

SmartTesting CarV4.6.4.1 - Experiment_CoAdS - TestPurposes (Library\booklet\Documents\SmartTesting\Projects_Fincau\ARMOUR\experimentation\workspaces\experimentE...

Project Preferences Help

Search tests

Steps

Default model instance

Include model instance

sensorInstance.sendRequest(RETRIEVE_FIRMWARE_UPDATE)

firmwareManagerInstance.receiveRequest()

firmwareManagerInstance.sendResponse(NEW_FIRMWARE_INVALID_SIGNATURE)

sensorInstance.receiveResponse()

sensorInstance.sendResponse(VALID_SIGNATURE)

firmwareManagerInstance.receiveResponse()

Point of view

Tags activated by the test (bold for current step)

AIM

DRPP_INVALID_SIGNATURE

FIRMWARE_MANAGER: FIRMWARE_REQUEST_RECEIVED

FIRMWARE_MANAGER: FIRMWARE_RESPONSE_SENT

FIRMWARE_MANAGER: RECEIVE_RESPONSE_FROM_SENSOR

FIRMWARE_MANAGER: SENDING_FIRMWARE_WITH_INVALID_SIGNATURE

SENSOR: SEND_FIRMWARE_INVALID_SIGNATURE

SENSOR: RECEIVE_RESPONSE_FROM_FIRMWARE

SENSOR: SEND_FIRMWARE_UPDATE_REQUEST

SENSOR: SEND_RESPONSE_TO_FIRMWARE

MD

function receiveRequest(charstring payload, integer v_id) runs on HType {

timer tc_ac := 10.0;

var ASP_UDP_message_v_receiveMessageBytes;

var CoAP_Message_v_receiveMessage;

var integer resp;

tc_ac.start;

alt {

HyPCO_PT.receive(v_receiveMessage(v_id)) -> value v_receiveMessageBytes

resp := f_CoAP_dec(v_receiveMessageBytes.data, v_receiveMessage);

if (match(v_receiveMessage, m_receivePayload(payload))){

set verdict(fail);

}; else {

set verdict(fail);

};

HyPCO_PT.receive {

set verdict(fail);

};

tc_ac.timeout {

set verdict(incon);

};

}

function sendRequest(charstring payload, integer v_id) runs on HType {

var octetstring v_sendMessageBytes;

var integer resp;

resp := f_CoAP_enc(valueof(m_sendPayload(payload)), v_sendMessageBytes);

HyPCO_PT.send(m_sendMessage(v_sendMessageBytes, v_id));

}

①

Vulnerability Analysis

②

Extracting API & Model Inference

③

Security test pattern selection

④

MBT test generation based on ARMOUR test strategies

⑤

Publication in TPlan – test description and TTCN-3 test scripts

Test results & Labelling analysis

Test Purpose Id	EXP1_ID6_01						
Test objective	Check that the Sensor successfully installs a firmware with a valid signature from the Firmware Manager (FM).						
Test Pattern Reference	TP_ID6						
Config Id	CST_01						
Stage	Boostrapping						
Initial conditions	with { the Sensor being in the "Initial state" }						
Expected behaviour	<table border="1"> <thead> <tr> <th>Test events</th> <th>Direction</th> </tr> </thead> <tbody> <tr> <td>when { the Sensor sends a valid RETRIEVE_FIRMWARE_UPDATE request to FM }</td> <td>FM ← Sensor</td> </tr> <tr> <td>then { the Sensor receives a NEW_FIRMWARE response from FM containing new Firmware with valid signature and the Sensor succeeds to validate the firmware signature and the Sensor sends a Response message containing Response Status Code set to INSTALLED_FIRMWARE }</td> <td>FM → Sensor FM ← Sensor</td> </tr> </tbody> </table>	Test events	Direction	when { the Sensor sends a valid RETRIEVE_FIRMWARE_UPDATE request to FM }	FM ← Sensor	then { the Sensor receives a NEW_FIRMWARE response from FM containing new Firmware with valid signature and the Sensor succeeds to validate the firmware signature and the Sensor sends a Response message containing Response Status Code set to INSTALLED_FIRMWARE }	FM → Sensor FM ← Sensor
Test events	Direction						
when { the Sensor sends a valid RETRIEVE_FIRMWARE_UPDATE request to FM }	FM ← Sensor						
then { the Sensor receives a NEW_FIRMWARE response from FM containing new Firmware with valid signature and the Sensor succeeds to validate the firmware signature and the Sensor sends a Response message containing Response Status Code set to INSTALLED_FIRMWARE }	FM → Sensor FM ← Sensor						

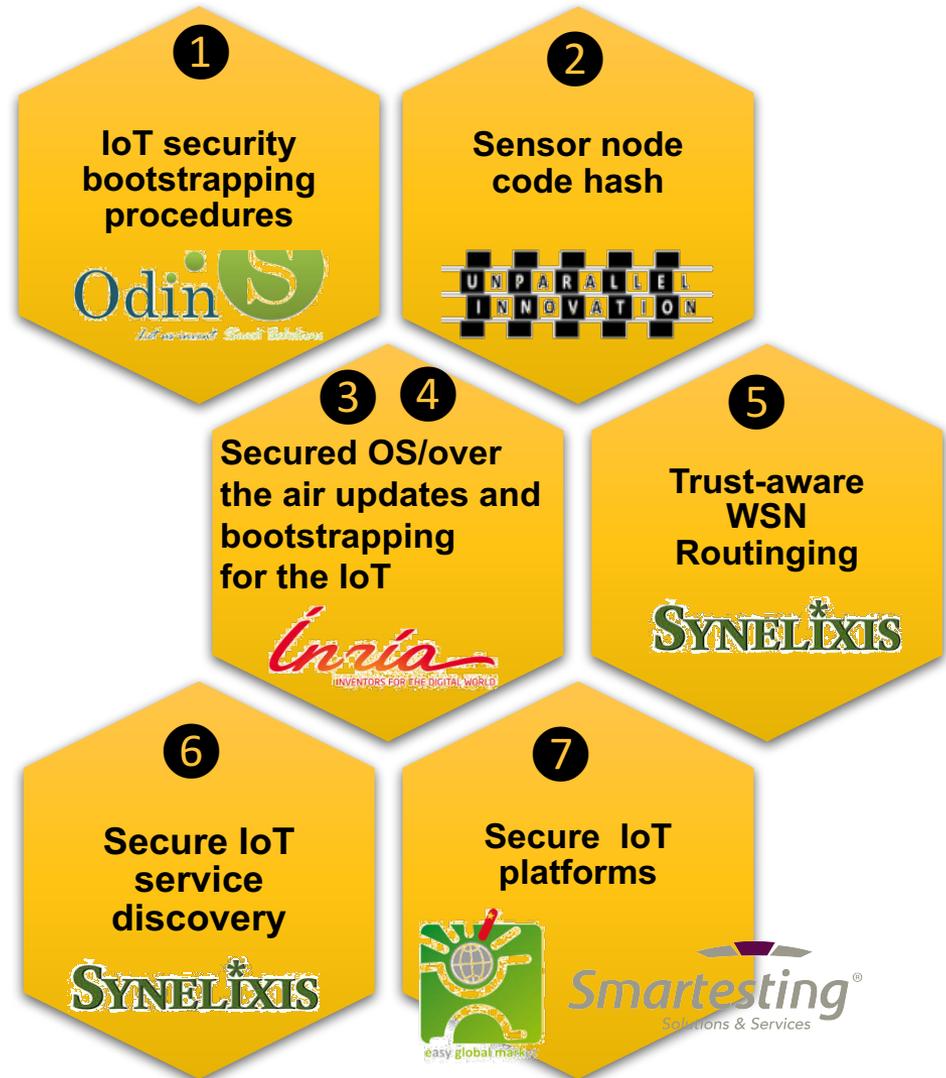
&

1. ARMOUR Project : Challenges and Objectives
2. Model-Based Security Testing
 - Approach & contribution for IoT Labelling and Certification
- 3. Experiments**
4. Conclusion

Seven IoT Security & Trust experiments

7 experiments covering complementary **segments of an IoT deployment:**

- Devices and data → 1, 2 & 3
- (Wireless) connectivity → 4 & 5
- Applications & Services → 6
- Platforms → 7



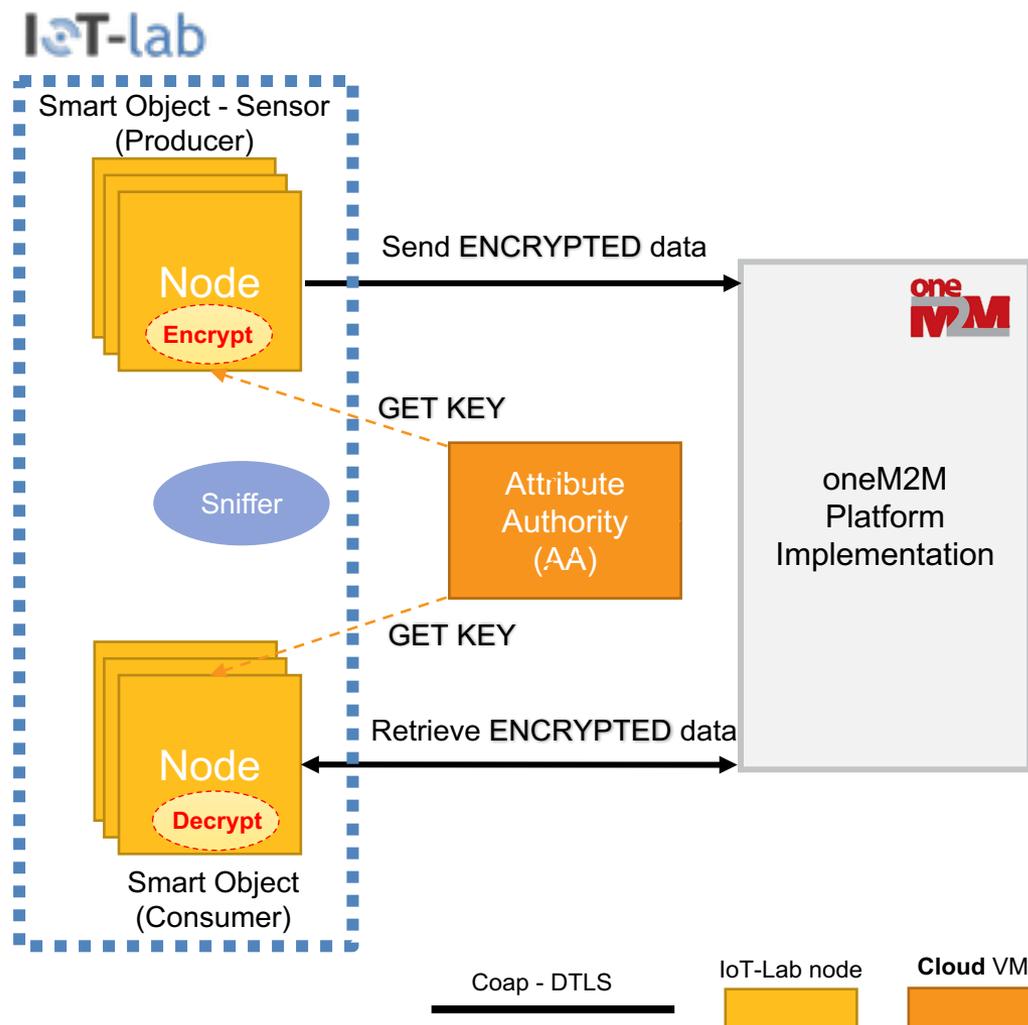
Test and validation of the framework



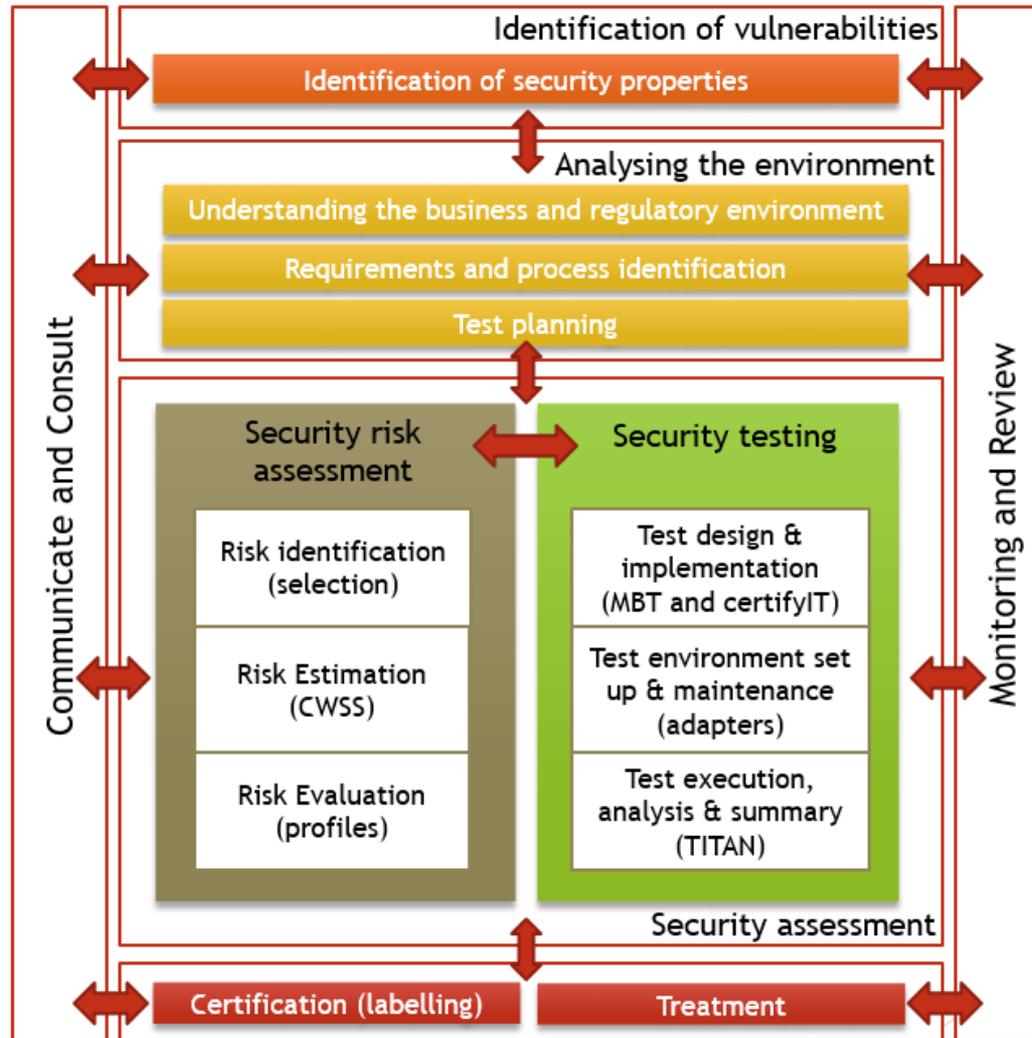
Contribution to testing campaigns

Large-Scale & End-2-End Case Study (Example)

TP ID	Security Test Patterns
TP_ID6	Run unauthorized software
TP_ID8	Resistance to eaves dropping and man in the middle
TP_ID10	Resistance to Injection Attacks
TP_ID11	Detection of flaws in authentication



Follow the ETSI approach and ISO 31000



Identification of vulnerabilities

- Database of general security threats in IoT (not included in ISO 31000)
- Compact threats of OneM2M to simplify and adapt to IoT devices also

Lack of Authentication

- The endpoints should be legitimate.

Replay attack

- Intermediate entity can store a data packet and replay it at a later stage.

Insecure cryptography

- The cryptographic suite and key length must be enough to avoid certain type of attacks, such as dictionary attack or force brute.

DoS attacks

- Several endpoints can access to the server at the same time in order to collapse it.

Lack of Integrity

- Received data are not tampered with during transmission; if this does not happen, then any change can be detected.

Lack of Confidentiality

- Transmitted data can be read only by the communication endpoints.

Lack of Authorization

- Endpoint services should be accessible to endpoints who have the right to access them.

Lack Fault tolerance

- Exceptions should be controlled to avoid faults that affects the endpoints.

Security testing



ARMOUR

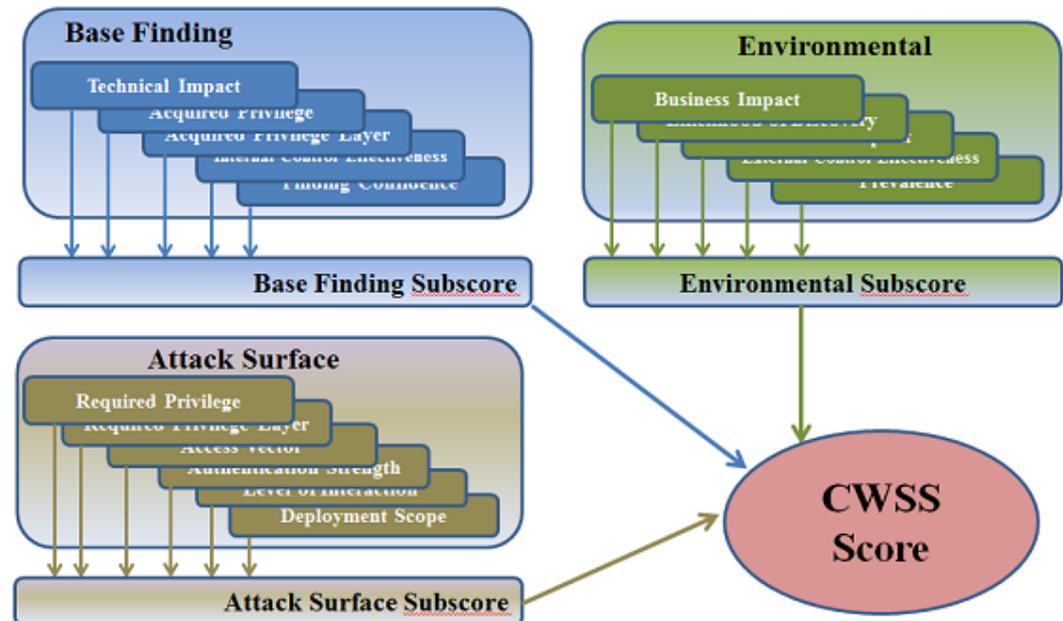
- From the vulnerabilities considered in the first phase, we produce security tests
- In the ETSI proposal, the automation of this phase is not contemplated, but the automation of this process eases the update of the label to cope with changing conditions in which the device operates.
- In ARMOUR project, this process is intended to be **automatized**:
- The tests provide us a series of security metrics such as % of ciphered data, % of messages protected against replay attack, etc. The different general tests are shown in the figure of mapping with OneM2M (relation).



Security risk assessment

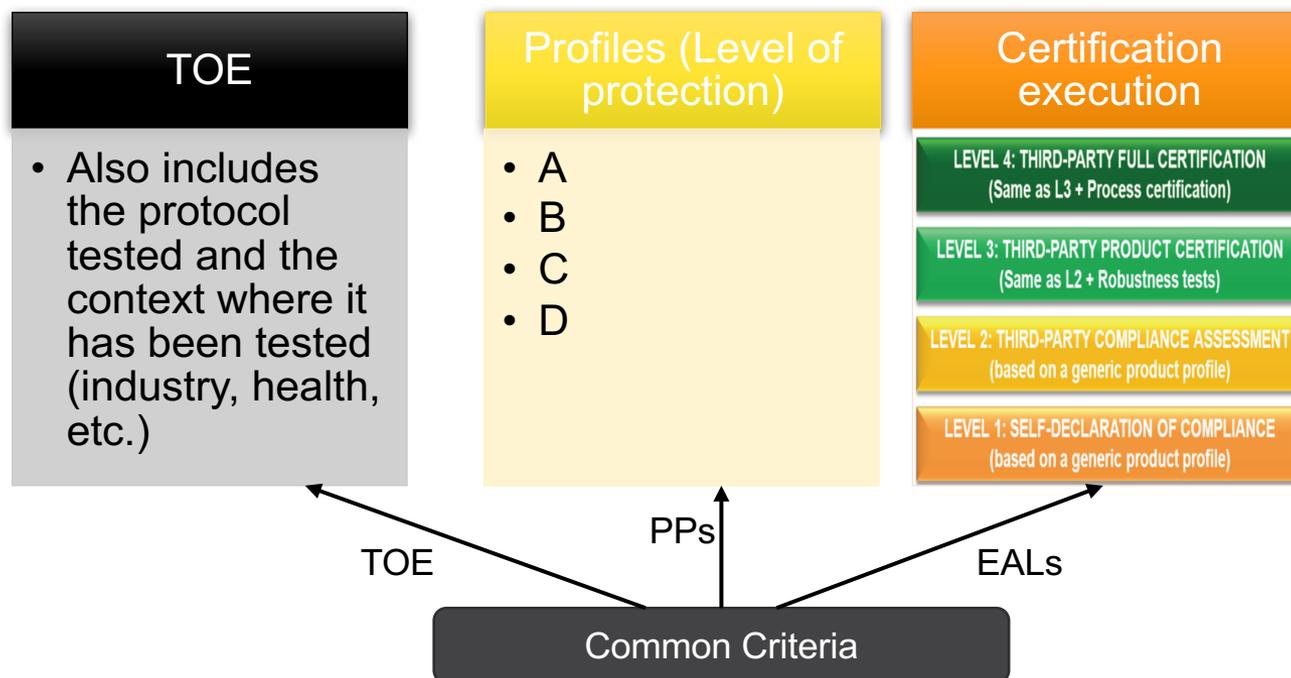
- The CWSS metrics can be obtained:
 - From testing
 - By default taking into account the vulnerability
 - By default if they are not applicable to IoT or to our certification procedure (e.g finding confidence, since the scenario is evaluated before being attacked) (*Risk Identification phase*)
- *Risk Estimation phase*: we calculate the score for each vulnerability by means of the CWSS formula:

$$S = BF * AS * ES$$



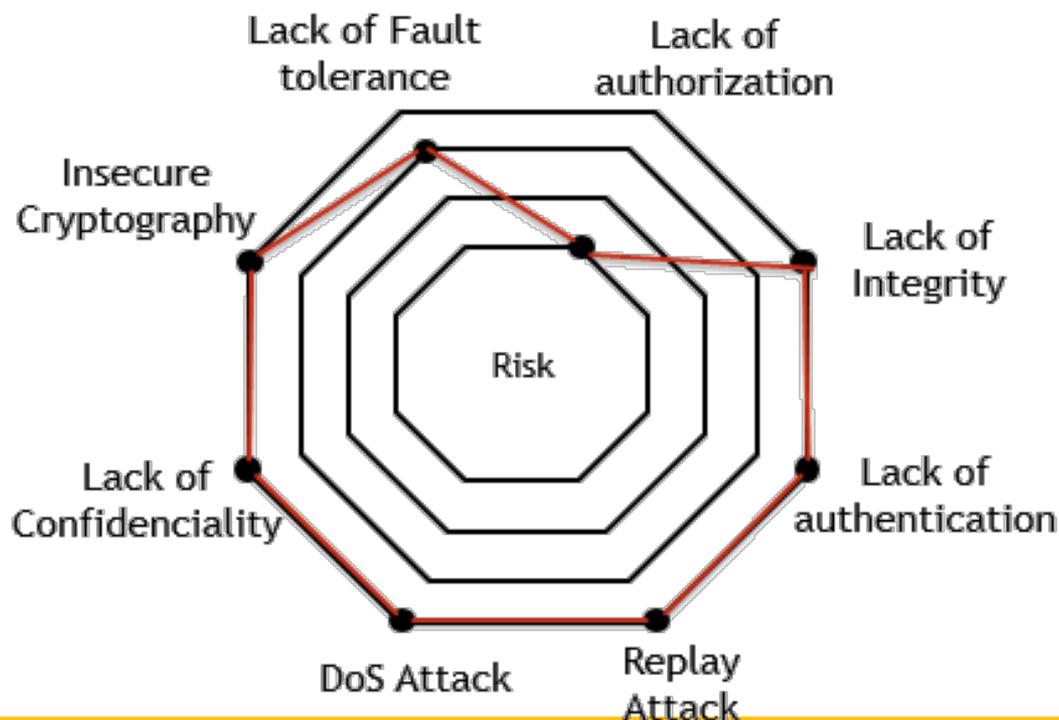
Certification - Labelling

- As an output of the general certification process, we obtain a label associated to the risk of the scenario tested.
- Three mains aspects are considered to be included in the label, following the Common Criteria approach



Certification - Labelling

- Visual labelling following the recommendations of ENISA and ECSO. The result of the evaluation need to be communicated appropriately to the user.
- Multidimensional, like security.
- To perform a fast labelling update, we propose the usage of a QR.



Conclusion : On the way to MBST for IoT Systems Labelling & Certification



ARMOUR

Security is number one challenge in the IoT domain



ARMOUR

- **Model-Based Security Testing as a core technology to ensure a trustable labelling scheme**
- **Domain-specific modeling**
- **Machine learning algorithms**



ARMOUR

Towards a Trust Label Process

supported by (large scale) IoT enhanced security test-beds



ARMOUR

Collaborations & contributions



IERC



AIOTI



oneM2M



ARMOUR



ARMOUR

www.armour-project.eu

ARMOUR - Large-Scale Experiments of IoT Security Trust

Call Identifier

H2020-ICT-2015

Topic

ICT-12-2015 Integrating experiments and facilities in FIRE+

Project Reference

688237

Type of Action

RIA - Research and Innovation action

Start date of the project

February 1st, 2016

Duration

24 months

