How hard can it be?

RIGHT?

Jarmo Lahtiranta, Senior Security Specialist

Jarmo Lahtiranta

- Security Consultant @ Insta Group - Finland
- Previously:
  - Ethical hacker
  - Finnish Cyber Security Centre - National CERT
  - Embedded / Product Security
- @naranek on Twitter and all around
- Disclaimer: All opinions & ramblings are my own and may or may not reflect the opinions of my employer

Fixing vulnerabilities

- Mitigate vulnerability / create a workaround
- Fix instance of vulnerability
- Fix vulnerability class
- Re-architect to remove vulnerability

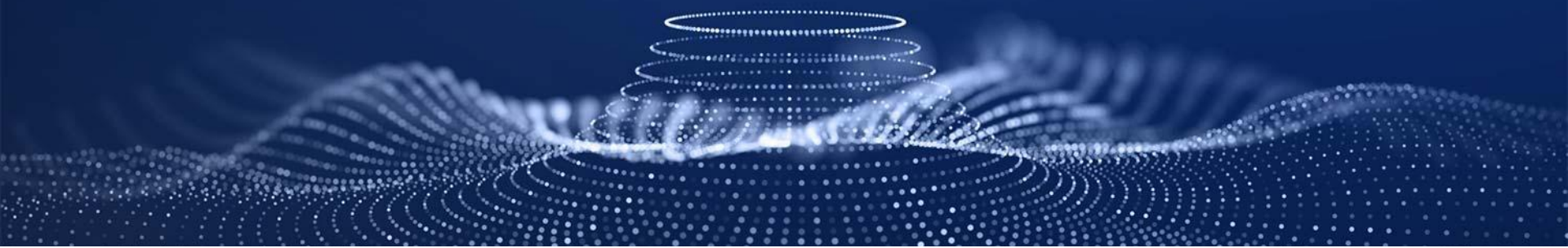
Fixing Default Credentials

- **Solution:** Create a password on first use
  - What if there's no UI?
- **Solution:** Have the device enroll on first use
  - Now anyone can enroll devices
- **Solution:** Preinstalled keys
  - Need to build the capability. Old devices won't have the keys.
  - You're edging closer to PKI land
  - Cryptography turns any problem into a key management problem

Hackers focus on breaking systems

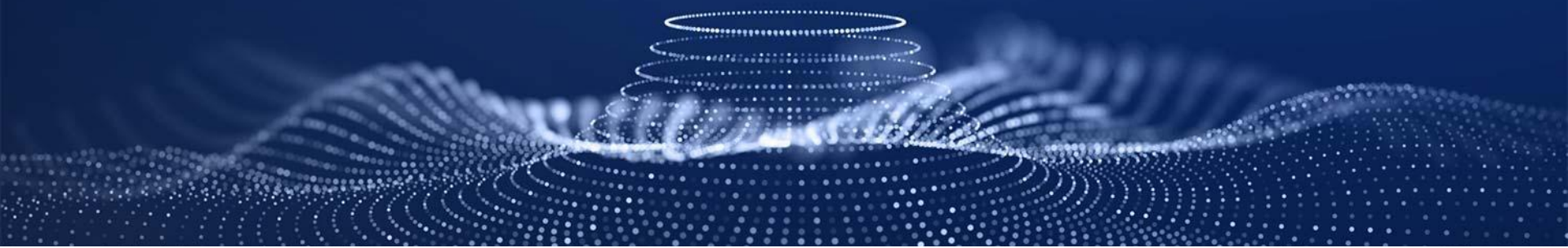
Developers focus on getting things to work

It’s not really a fair fight



“Attackers only need to succeed once. Defenders need to succeed every time.”

Not really, but…



Fixing this vulnerability is a top priority

So are these other 10 things 😩

You could go out of business if you are breached

Yeah but we'll definitely go out of business if we can't get this product out the door right now

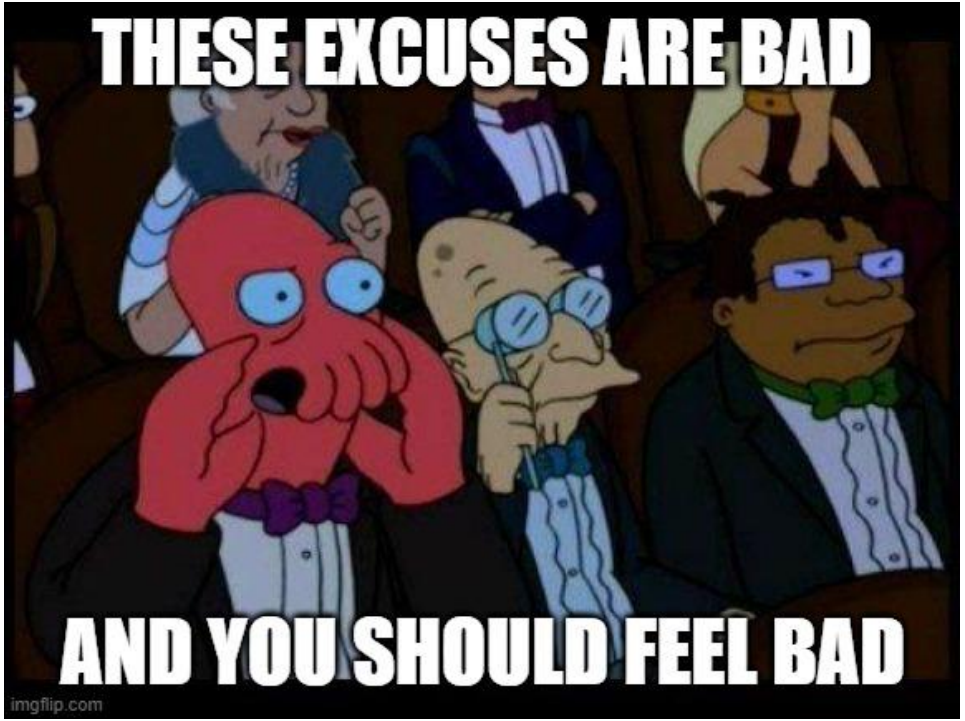
Hackers
don't give a shit:

New Zealand's Hacker con - Wellington

- [ ] About your project's scope
- [ ] It's managed by a third party
- [ ] It's a legacy system
- [ ] It's "too critical to patch"
- [ ] About your outage windows
- [ ] About your budget
- [ ] You've always done it that way
- [ ] About your Go-Live Date
- [ ] It's only a pilot proof of concept
- [ ] About Non-Disclosure Agreements
- [ ] It wasn't a requirement in the contract
- [ ] It's an internal system
- [ ] It's really hard to change
- [ ] It's due for replacement
- [ ] You're not sure how to fix it
- [ ] It's handled in the Cloud
- [ ] About your Risk Register entry
- [ ] The vendor doesn't support that configuration
- [ ] It's an interim solution
- [ ] It's [insert standard here] compliant
- [ ] It's encrypted on disk
- [ ] The cost benefit doesn't stack up
- [ ] "Nobody else could figure that out"
- [ ] You can't explain the risk to "The Business"
- [ ] You've got other priorities
- [ ] About your faith in the competence of your internal users
- [ ] You don't have a business justification
- [ ] You can't show Return on Investment
- [ ] You contracted out that risk

Attackers don’t care about…

- About Non-Disclosure Agreements
- About your faith in the competence of your internal users
- It's [insert standard here] compliant
- "Nobody else could figure that out"
- It's only a pilot/proof of concept
- It's an internal system
- It's an interim solution
- You've always done it that way
- It's handled in the Cloud
- It's a legacy system



Risk Accepted

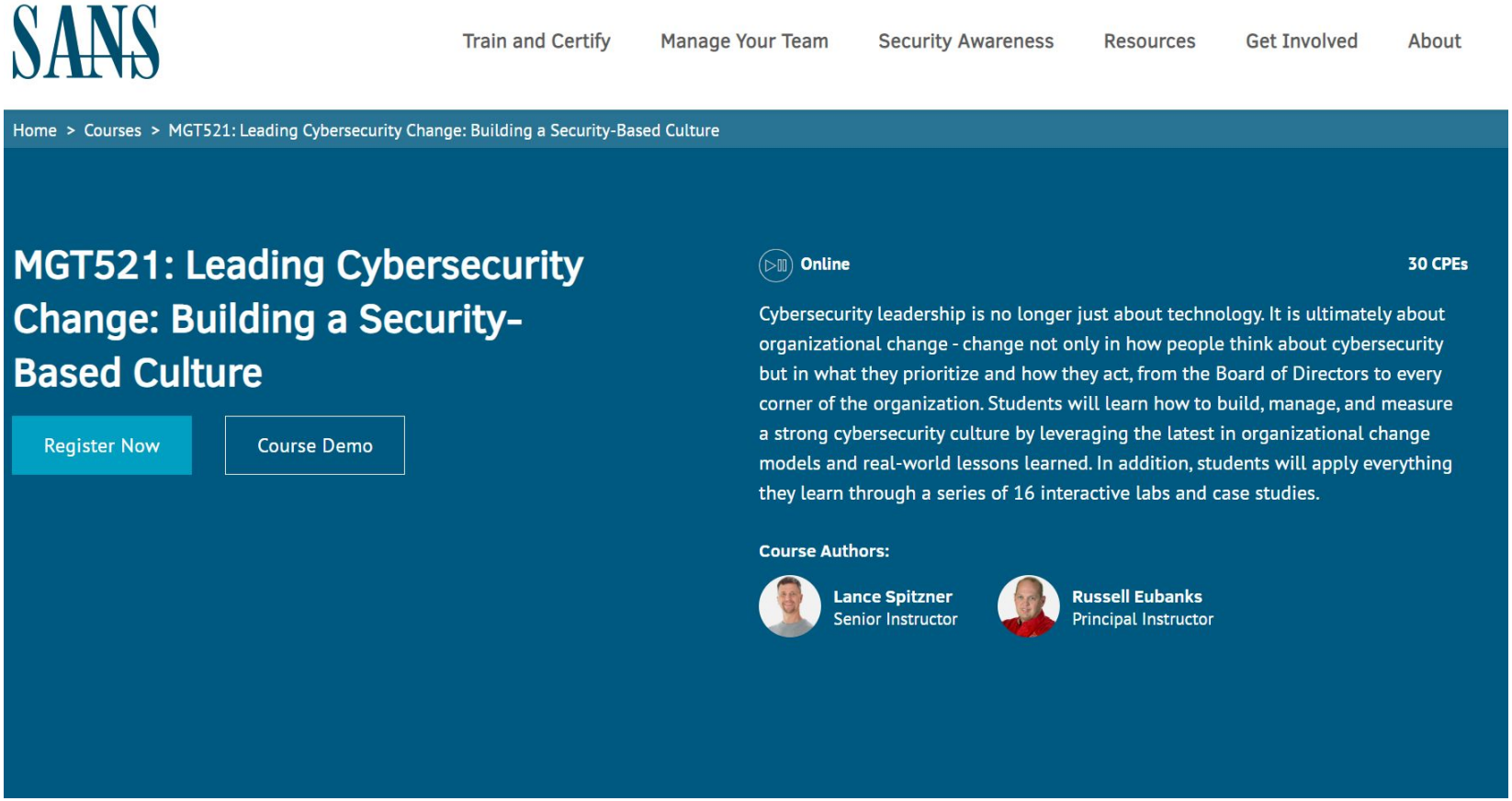
- [ ] Its due for replacement
- [ ] It's "too critical to patch"
- [ ] About your Go-Live Date
- [ ] About your outage windows

Help Needed

- [ ] It's really hard to change
- [ ] You're not sure how to fix it

Business issues

- [ ] About your budget
- [ ] You dont have a business justification
- [ ] You've got other priorities
- [ ] You can't explain the risk to "The Business"
- [ ] You can't show Return on Investment
- [ ] The cost benefit doesn't stack up
- [ ] It wasn't a requirement in the contract
- [ ] About your project's scope



https://www.sans.org/cyber-security-courses/leading-cybersecurity-change/

Most people want to do a good job

Carrot is mightier than the stick

Building trust is essential

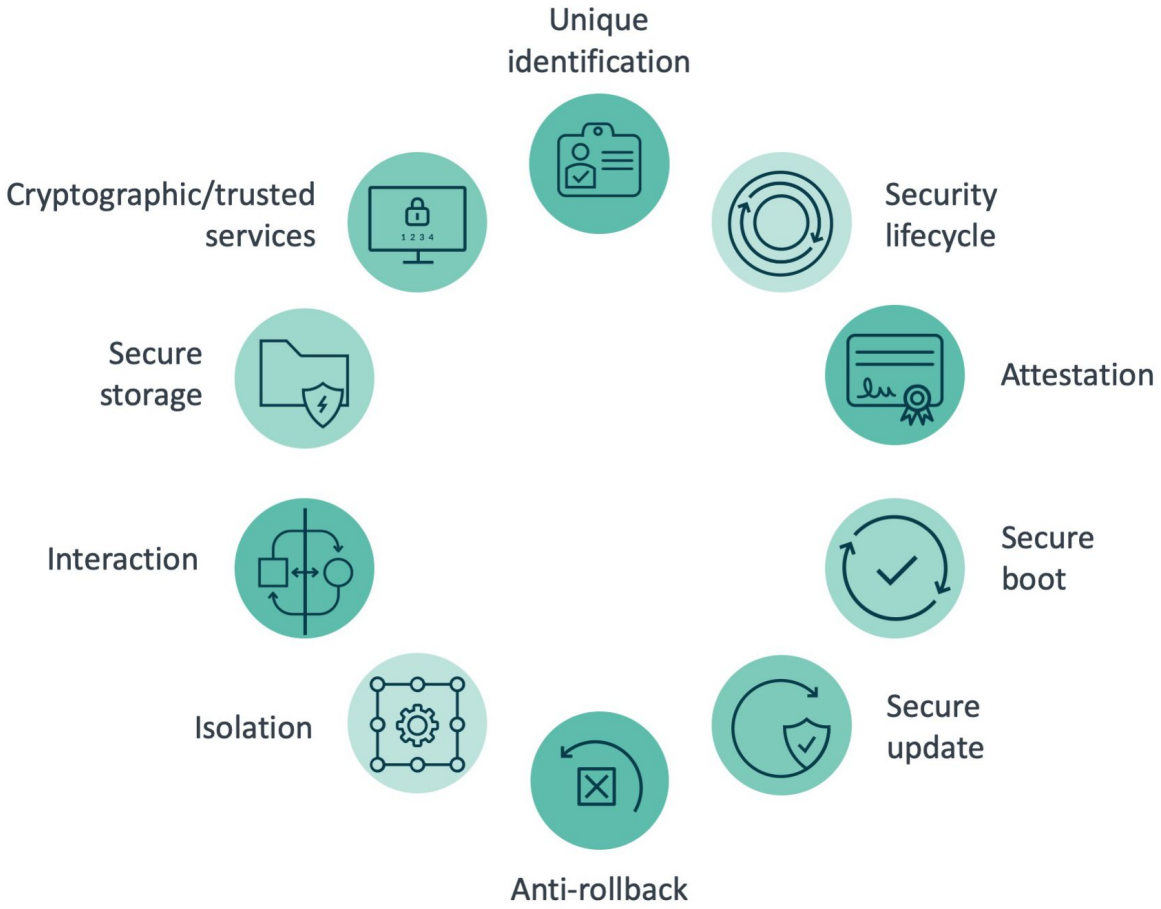
Why are IoT devices insecure?

Old hardware doesn’t support security features

But we have TrustZone for Cortex-M & -A etc.

We still use older chips because they are cheaper

It’s too costly to use them and we don’t even know how


Unique identification
Security lifecycle
Attestation
Secure boot
Secure update
Anti-rollback
Isolation
Interaction
Secure storage
Cryptographic/trusted services
1 2 3 4

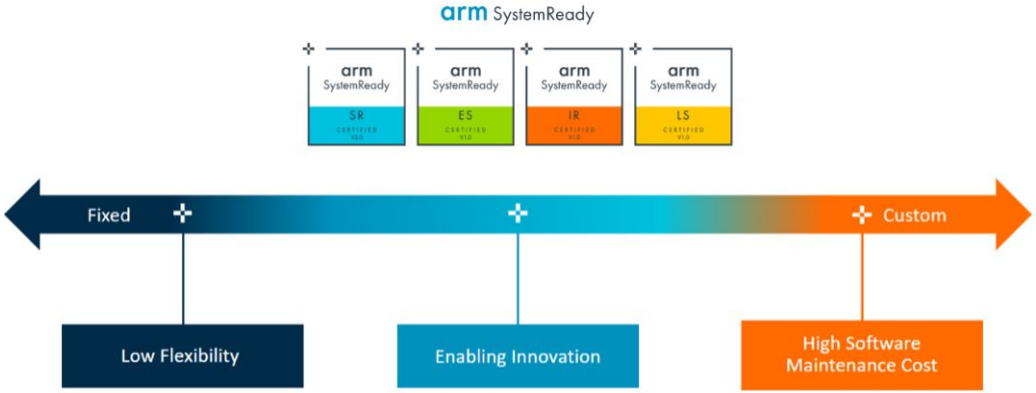
You should use a device management framework to take care of the security of your platform

We don’t want to pay for that

We’ll build our own solution instead

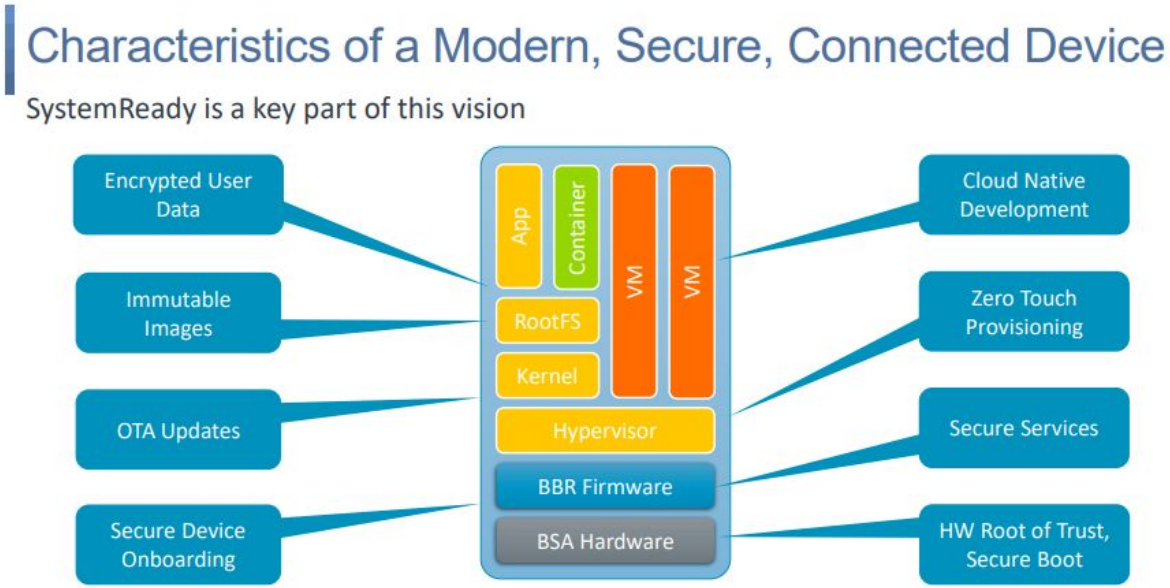
ARM SystemReady IR

- SystemReady IR-certified platforms implement a minimum set of hardware and firmware features that an operating system can depend on to deploy the operating system image.
- SystemReady IR is tailored to meet the needs of embedded Linux or BSD ecosystem on systems based on embedded Arm SoCs.



https://developer.arm.com/Architectures/Arm%20SystemReady%20IR

ARM SystemReady IR



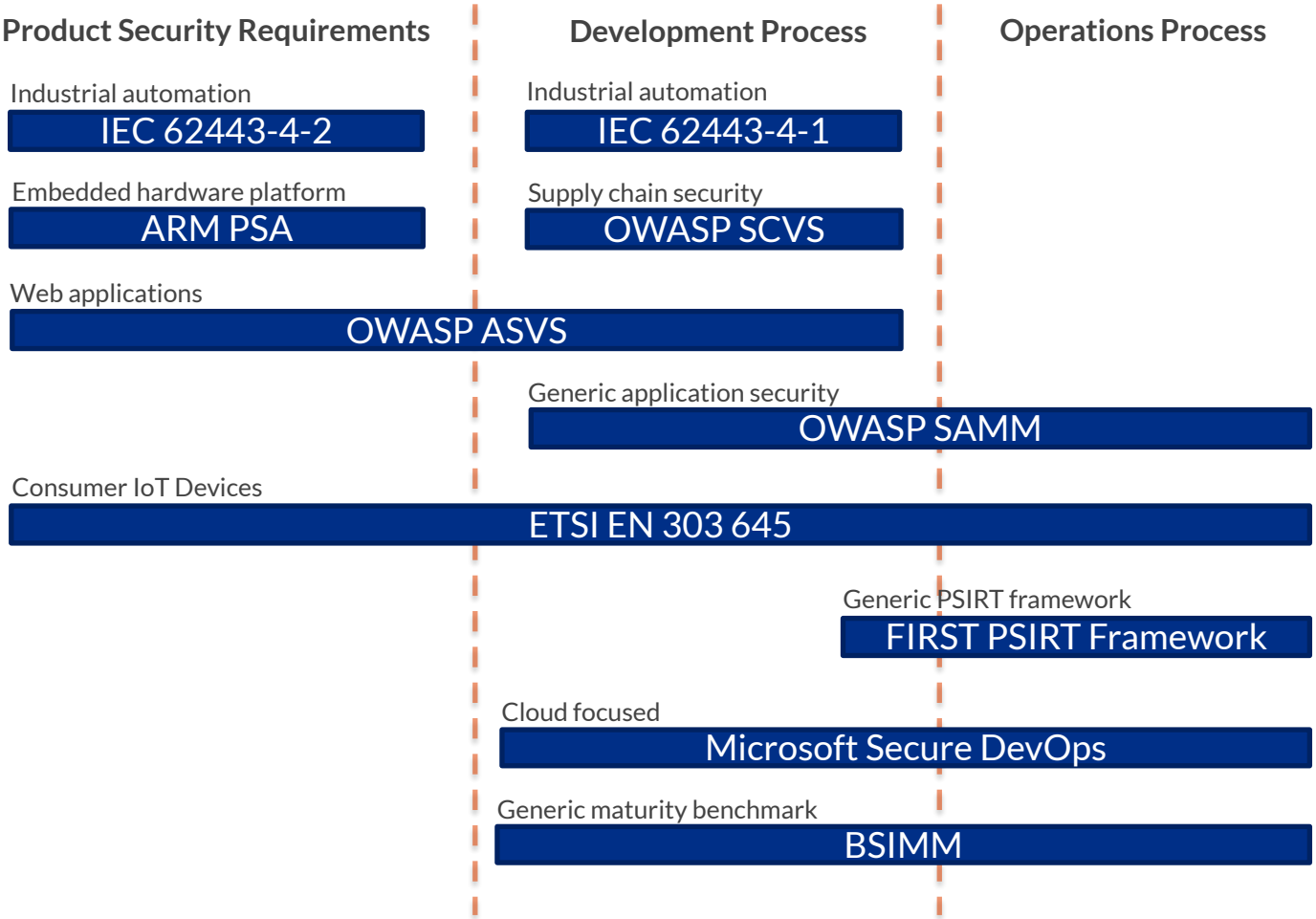
https://resources.linaro.org/en/resource/eXaLL5EumNCKbBZW9Rd3pj

“You’re in charge of securing this product”

What do you do?

STANDARDS

THIS SHOULD BE FUN...


Product Security Requirements
Development Process
Operations Process
Industrial automation
IEC 62443-4-2
Industrial automation
IEC 62443-4-1
Embedded hardware platform
ARM PSA
Supply chain security
OWASP SCVS
Web applications
OWASP ASVS
Generic application security
OWASP SAMM
Consumer IoT Devices
ETSI EN 303 645
Generic PSIRT framework
FIRST PSIRT Framework
Cloud focused
Microsoft Secure DevOps
Generic maturity benchmark
BSIMM

Standards & Frameworks

- FIRST PSIRT Framework
  - The Services Frameworks are high level documents detailing possible services that computer incident response teams (CSIRTs) and product incident response teams (PSIRTs) may provide.
- OWASP ASVS – Application Security Verification Standard
  - Provides basis for testing **web application technical security controls** and also provides developers with a list of **requirements for secure development**.
- OWASP SCVS – Software Component Verification Standard
  - Framework for identifying activities, controls, and best practices, which can help in identifying and **reducing risk in a software supply chain.**
- OWASP SAMM – Software Application Maturity Model
  - Our mission is to provide an effective and measurable way for you to analyze and improve your **secure development lifecycle**.
- BSIMM – Building Security In Maturity Model
  - Building Security In Maturity Model (BSIMM) is a study of current software security initiatives or programs. It quantifies the application security (appsec) practices of different organizations across industries, sizes, and geographies while identifying the variations that make each organization unique.

Standards & Frameworks

- ARM PSA - Platform Security Architecture
  - 10 security goals for creating a safe embedded platform to build on
  - Close to hardware
- ETSI EN 303 645: Consumer IoT Cyber Security Baseline Requirements
  - 13 security requirements

ASVS – Application Security Verification Standard

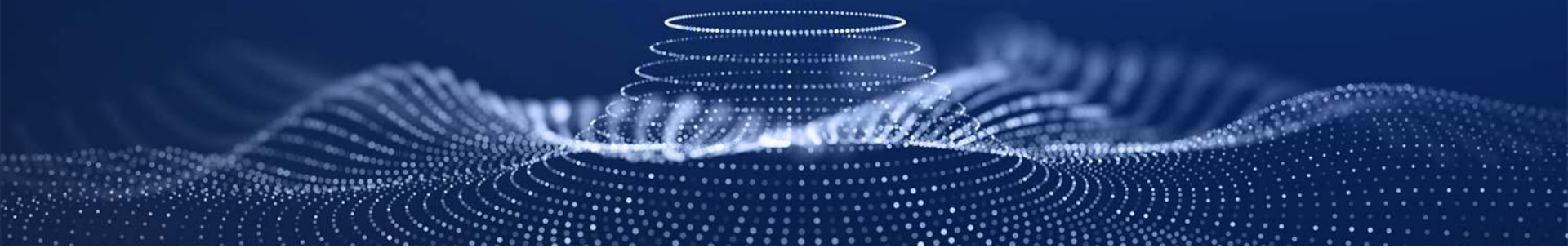
- 71 pages
- 7 Chapters
- 278 Controls in total
  - 128 on Level 1
  - 132 on Level 2
  - 19 on Level 3

- **Level 1 -** First steps, automated, or whole of portfolio view
  - “Level 1 is the bare minimum that all applications should strive for.”
- **Level 2** - Most applications
  - “An application achieves ASVS Level 2 (or Standard) if it adequately defends against most of the risks associated with software today.”
- **Level 3** - High value, high assurance, or high safety
  - “This level is typically reserved for applications that require significant levels of security verification, such as those that may be found within areas of military, health and safety, critical infrastructure, etc.”

You have a team of 3 developers

The standard has 278 security controls

What do you do?



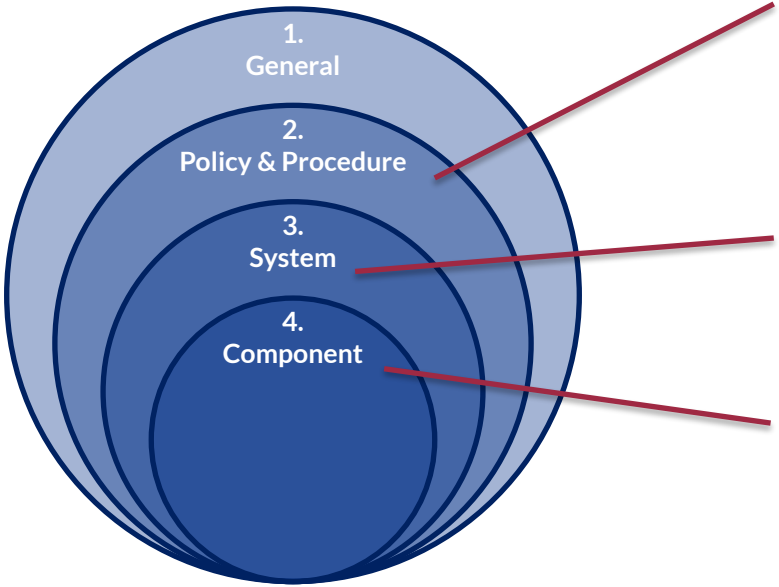
OWASP Security Levels

- Software Components (OWASP SCVS)
  - **SCVS Level 1** is for low-assurance requirements where basic forms of analysis would suffice.
  - **SCVS Level 2** is for moderately sensitive software where additional analysis or due diligence is required.
  - **SCVS Level 3** is for high-assurance requirements due to the sensitivity of data or use of the software

- Application Security (OWASP ASVS)
  - **ASVS Level 1** is for low assurance levels, and is completely penetration testable
  - **ASVS Level 2** is for applications that contain sensitive data, and is **the recommended level for most apps**
  - **ASVS Level 3** is for the most critical applications - applications that perform high value transactions, contain sensitive **medical data**, or any application that requires the highest level of trust.

- Mobile Application Security (OWASP MASVS)
  - **MASVS-L1:** Application adheres to mobile application security best practices. This level is appropriate for **all mobile applications**.
  - **MASVS-L2:** This level is appropriate for apps that handle **highly sensitive data**, such as mobile banking apps.
  - **MASVS-R:** MASVS-R is applicable to apps that handle highly sensitive data and may serve as a means of **protecting intellectual property or tamper-proofing** an app.

IEC 62443 - Security for Industrial Automation and Control Systems (IACS)

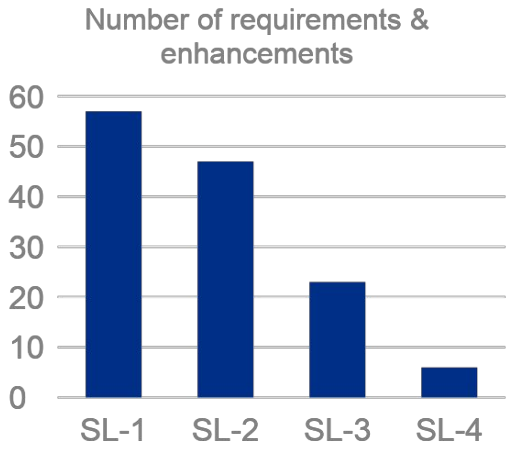


2: Applies to Asset Owners
2-1: Requirements for an IACS security management system
2-2: Implementation guidance
2-3: Patch management
2-4: Installation & maintenance

3: Applies to System Integrators
3-1: Security technologies for IACS
3-2: Security levels for zones and conduits
3-3: System security requirements and levels

4: Applies to Component Suppliers
4-1: Product development requirements
4-2: Technical security requirements for IACS components

Security Levels in 62443-4-2



Prevent the unauthorized disclosure of information:

- **SL 1** – Via eavesdropping or casual exposure.

…to an entity actively searching for it using:

- **SL 2** – Simple means with low resources, generic skills and low motivation.
- **SL 3 – Sophisticated** means with **moderate** resources, **IACS specific skills** and **moderate** motivation.
- **SL 4** – Sophisticated means with **extended** resources, IACS specific skills and **high** motivation.

Embedded Device Requirements

SL 1 - Integrity

- Controlled execution of mobile code (Java, JS, PS, ActiveX etc.)
- Protection from malicious code (by for example code signing)
- Support for updates
- Integrity of the boot process

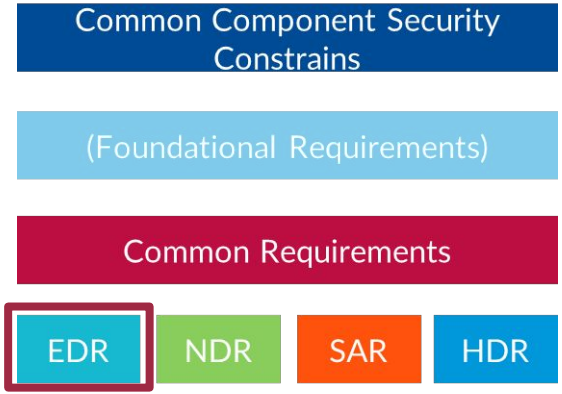
SL 1 – Via eavesdropping or casual exposure.

Embedded Device Requirements

SL 2 - Authenticity

- Controlled execution **and authenticity checking** of mobile code
  - **Separate root-of-trust keys for mobile code stored securely (EDR 3.13)**
- Protection from malicious code by for example code signing
- Support **and authenticity checking** for updates
- Integrity **and authenticity** of the boot process **(Secure boot)**
- **Disabling physical diagnostic and test interfaces**
- **Hardware root-of-trust**

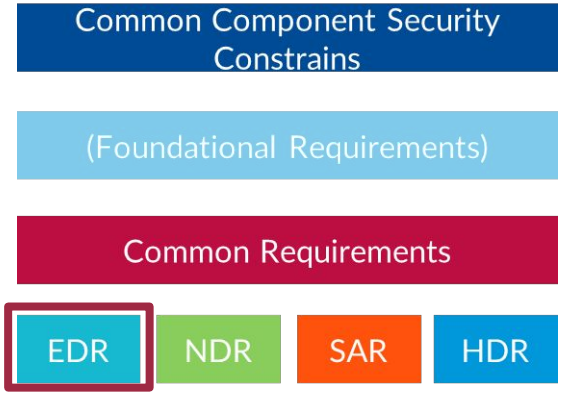
SL 2 – Simple means with low resources, generic skills and low motivation.



Embedded Device Requirements

SL 3 – Active monitoring

- Previous requirements
- **Active monitoring and logging of diagnostics & test interface access**
- **Automatic notification of unauthorized physical access**

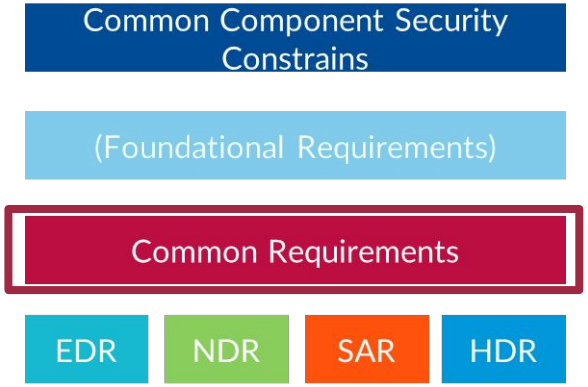


SL 3 – Sophisticated means with **moderate** resources, **IACS specific skills** and **moderate** motivation.

Embedded Device Requirements

SL 4

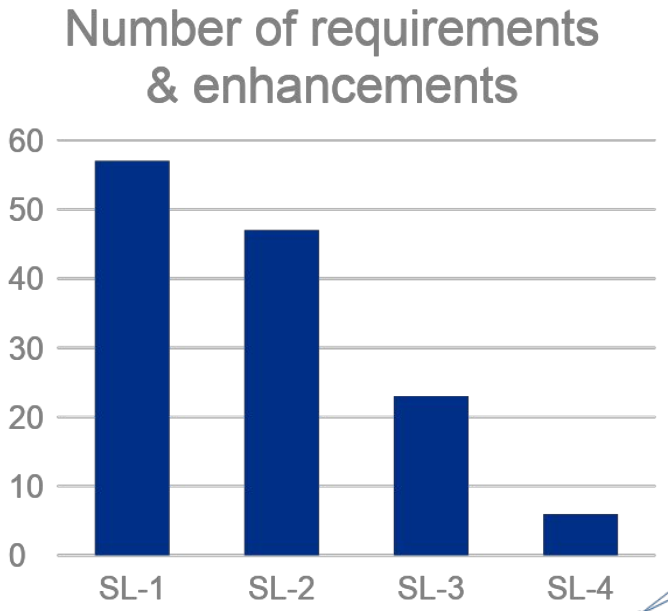
- No extra Embedded Device Requirements
- New Common Requirements
  - Password lifetime restrictions for all users (human, software process, or device)
  - Dual approval
  - Protection of time source integrity
  - Non-repudiation for all users
  - Security functionality verification during normal operation
  - Audit records on write-once media



SL 4 – Sophisticated means with **extended** resources, IACS specific skills and **high** motivation.

Security Requirements in 62443-4-2

- Topics
  - Identification and authentication control
  - Use control
  - System integrity
  - Data confidentiality
  - Restricted data flow
  - Timely response to events
  - Resource availability
- Device specific requirements
  - Software application requirements
  - Embedded device requirements
  - Host device requirements
  - Network device requirements



- *"All of the components defined in this document shall be developed and supported following the **secure product development processes** described in IEC 62443-4-1."*

62443-4-1 – Secure Development Lifecycle

46 requirements in total

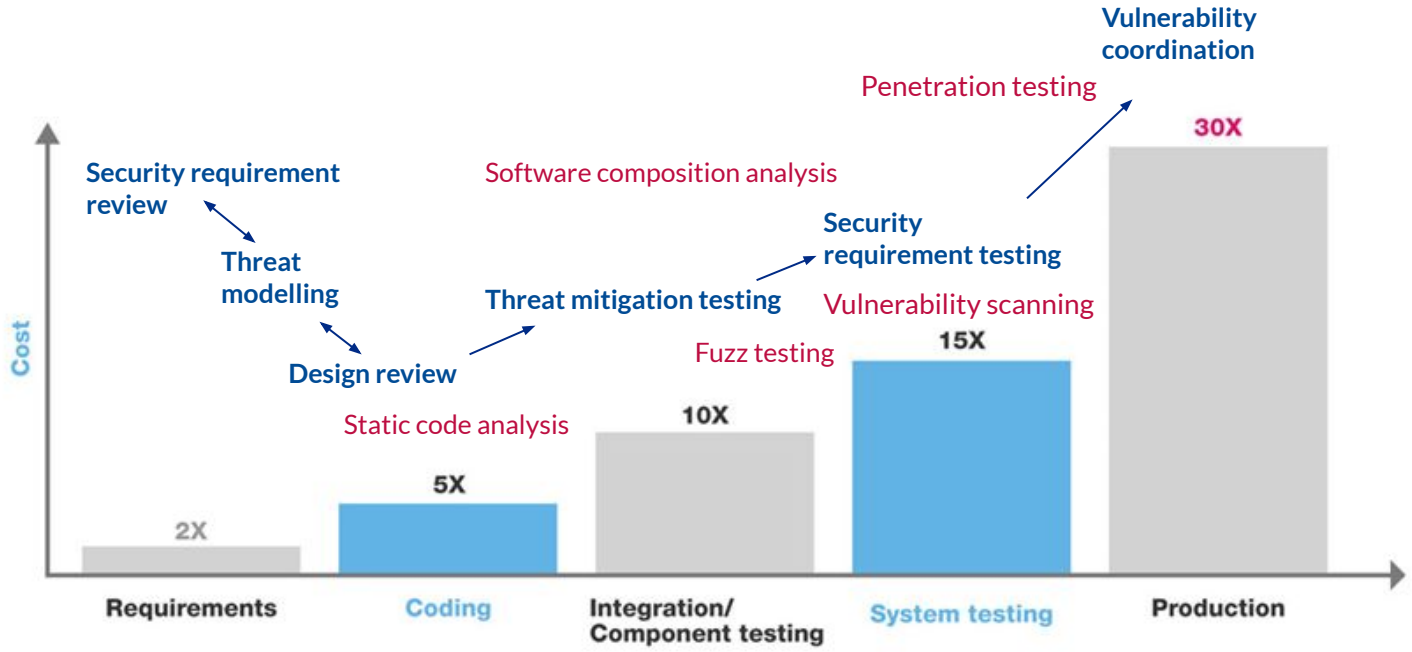
- **13:** Security management
- **5:** Specification of security requirements
- **4:** Secure by design
- **2:** Secure implementation
- **5:** Security verification and validation testing
- **6:** Management of security-related issues
- **5:** Security update management
- **6:** Security guidelines


J Wolfgang Goerlich
@jwgoerlich
Replying to @madplatt
The thing I love about maturity levels?
You can have a proactive process. You can have the process defined. You can have it documented and have people trained. You can have maturity defined and well on the way towards managed.
And it can still be a stupid process.
8:07 PM · Apr 23, 2022 · Twitter Web App

Cost of fixing vulnerabilities

Reactive methods

Proactive methods



Source: NIST (National institute of standards and techn.)

What does it take to build secure products?

- Security requirements
- Secure development process
- In all phases of production
- Starting from sales
- Where security is priced in

How much security is enough?

It depends on

Threats you're facing

Threats your products or customers are facing

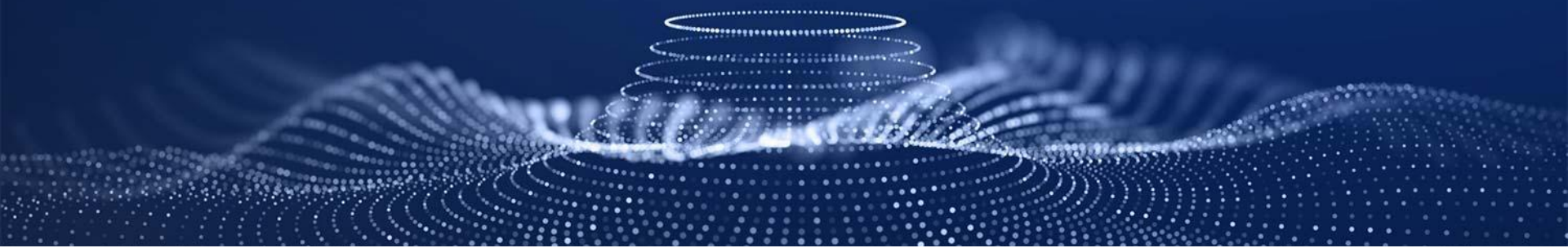
How much you or your customers are willing to pay

How much the attackers have to gain

Security has a cost

It needs to be priced separately

Customers have to demand it & be willing to pay for it



We know how secure products

The technology & standards are there

The future is looking brighter

Most people want to do a good job

Building trust is essential

References

- Arm SystemReady IR - https://developer.arm.com/Architectures/Arm%20SystemReady%20IR
- Arm PSA Security Goals - https://www.psacertified.org/blog/psa-certified-10-security-goals-explained/
- OpenSSF Scorecard - https://github.com/ossf/scorecard
- SLSA - https://slsa.dev
- OWASP ASVS - https://owasp.org/www-project-application-security-verification-standard/

Q&A?

Jarmo Lahtiranta - @naranek

jarmo.lahtiranta@insta.fi

insta.fi