TOWARDS A SECURE IOT LANDSCAPE

DANNY DE COCK

HTTP://GODOT.BE/SLIDES

March 2017

LIMITED SCOPE ☺

- Security challenges for commonly available and used devices
- IoT device not different from any other IT device
  - Communicates with its surroundings
  - Firmware, operating system, applications, application data, user data, configurations
- Happy when it works
  - Do not touch/reconfigure a working system ☺
    - Limited management of keys, algorithms, protocols, credentials…
  - Backward compatibility constricts deployment of secure environments
  - Everybody believes he/she is a cryptographer ☹
  - Very primitive key management
  - User’s lack of security awareness

STRAIGHTFORWARD OBSERVATIONS

- IoT focuses on functionality, locking-in a client, no focus on security
  - Security is afterthought after having secured the client
- Each family of devices works in its own silo
  - Aggregation of isolated component groups rather than integration
- User data, preferences & behavior immediately pushed to cloud services
  - Who manages the cloud, who is it and where can you find them?
  - User awareness: end-user has no insight about what happens to her data
- Authentication, confidentiality and authorization problems
  - Silo-based management of keys, preferences, access control settings…
  - No real key management for individual instantiations
  - Low power = lightweight communications and security protocols

“OUR SYSTEM IS SECURE: WE USE THE AES”

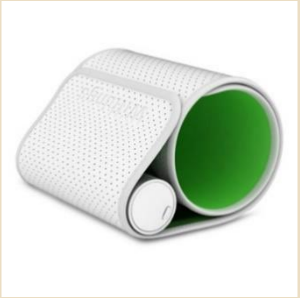
- What about
  - Key management
    - “Random” keys?
    - Authenticated (?) key agreement
  - Implementation
    - Modes of encryption, initialization vectors,…
    - Attacking the implementation
- Who holds the keys?
  - Who can use the keys?
  - Stored in the clear?
  - Key archives?

IOT SECURITY PROTOCOLS

- Protocols derived on well known classic protocols, e.g., TLS
  - Giving developers more choice can lead to security vulnerabilities
- Algorithms typically used:
  - Asymmetric: RSA, DSA/DHE, ECDSA, ECDHE
  - Symmetric encryption: AES, AES-CCM, AES-GCM
  - Symmetric authentication: AES-CCM, HMAC-SHA1/2/3
- Current IoT protocols use default algorithms
  - AllJoyn – open source, AllSeen Alliance – Qualcomm, Microsoft, AT&T…
  - Iotivity – open source, Open Interconnect Consortium – Intel, Samsung, Cisco…
  - Thread – open protocol, Thread Group – ARM, Samsung, Qualcomm…

THE INTERNET OF EVERYTHING

- Things
  - Controlled devices
  - Sensors
  - Monitors
  - Control points
  - Appliances
  - Wearables & washables ☺
- Remote controllers
  - ‘Personal’ control
  - Location & behavior
- Manufacturers
  - Updates & control
- Meta controllers, e.g.,
  - Fully automated scenarios



Similar functionalities: NEST, NXP, WIGWAG… Images: www.audi.com, www.belkin.com, www.fitbit.com , www.ifttt.com, www.nest.com, www.telenet.be, www.withings.com

THINGS, DATA, SERVICES, CONTROL – USER VIEW

- Users want
  - Free services
  - Maximum convenience
  - Maximum simplicity
- But
  - Forced harvesting of user data & settings
    - No user-awareness or concern
  - All data stored in the cloud
    - No user-transparency
    - No do-it-yourself-configuration possibilities
  - Free services come with promises
    - No guarantees
    - No commitments

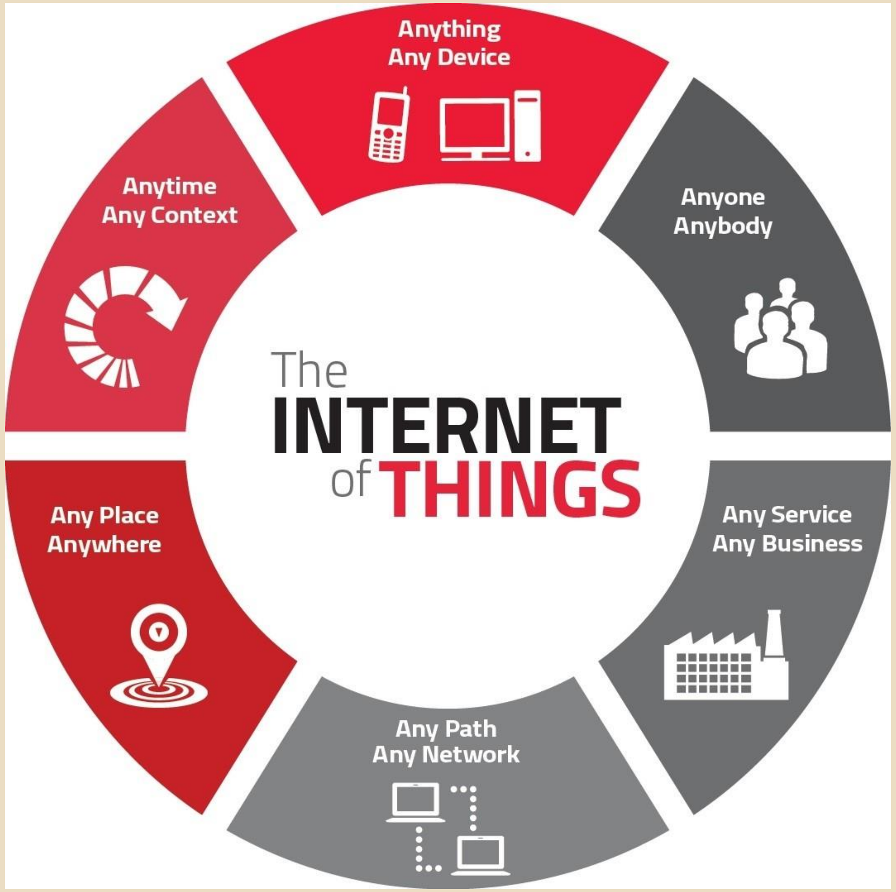


Image: www.informationsecuritybuzz.com

BENEFITS OF SECURE SOFTWARE DEVELOPMENT

- Application security
  - Important emerging requirement in software development
    - It is expected… no longer explicitly required
  - Controls potential
    - Severe brand damage
    - Financial loss
    - Privacy breaches
- Risk-aware customers (financial institutions, governmental organizations) want to
  - Assess the security posture of products they build or purchase
  - Plan to ultimately hold vendors accountable for security problems in their software
  - Procure reliable and secure software
    - Hold vendors accountable for security problems in software

CORE (IOT) SECURITY PROBLEMS

- Software development lifecycle does not deal well with security
  - Software developers lack structured guidance
  - Books on the topic are
    - Relatively new
    - Collections of unrelated good practices
- Security is not a feature that demos well
  - Developers tend to focus on core functionality features
- Security is addressed ad hoc by developers
  - Developers typically provide a minimal set of security services given their limited security expertise
- Applications are too complex to comprehend

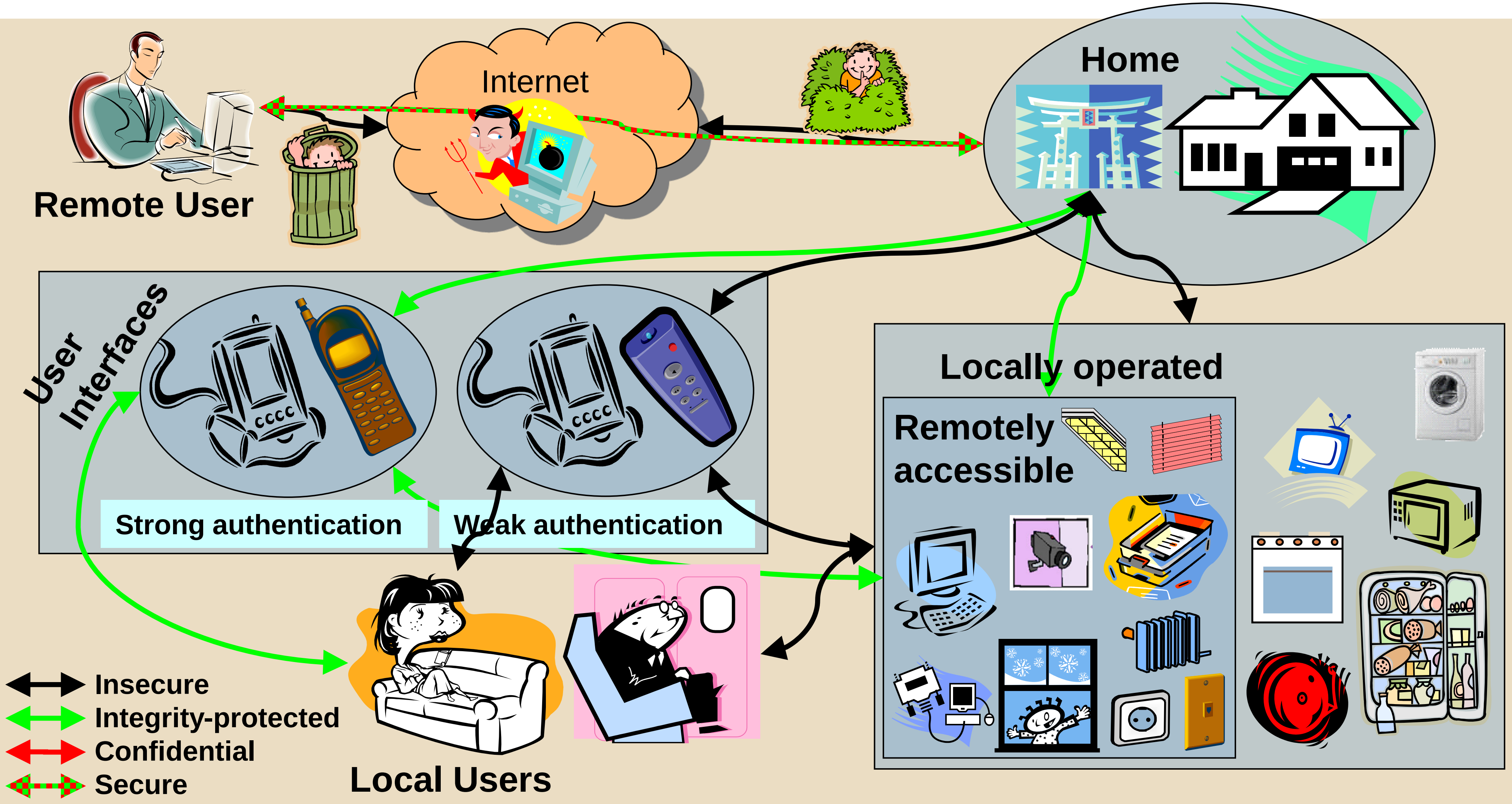
SECURE VS. SECURITY SOFTWARE

- Secure software
  - Application acts according to its specifications
  - Provable features of the application
  - Software design is the bottleneck
- Security software
  - Relies on secure software
  - Application uses secret and private information
    - Electronic payments, voting, signing,…
    - Protection of privacy, confidentiality, integrity,…
  - Critical use of the user/device/… credentials

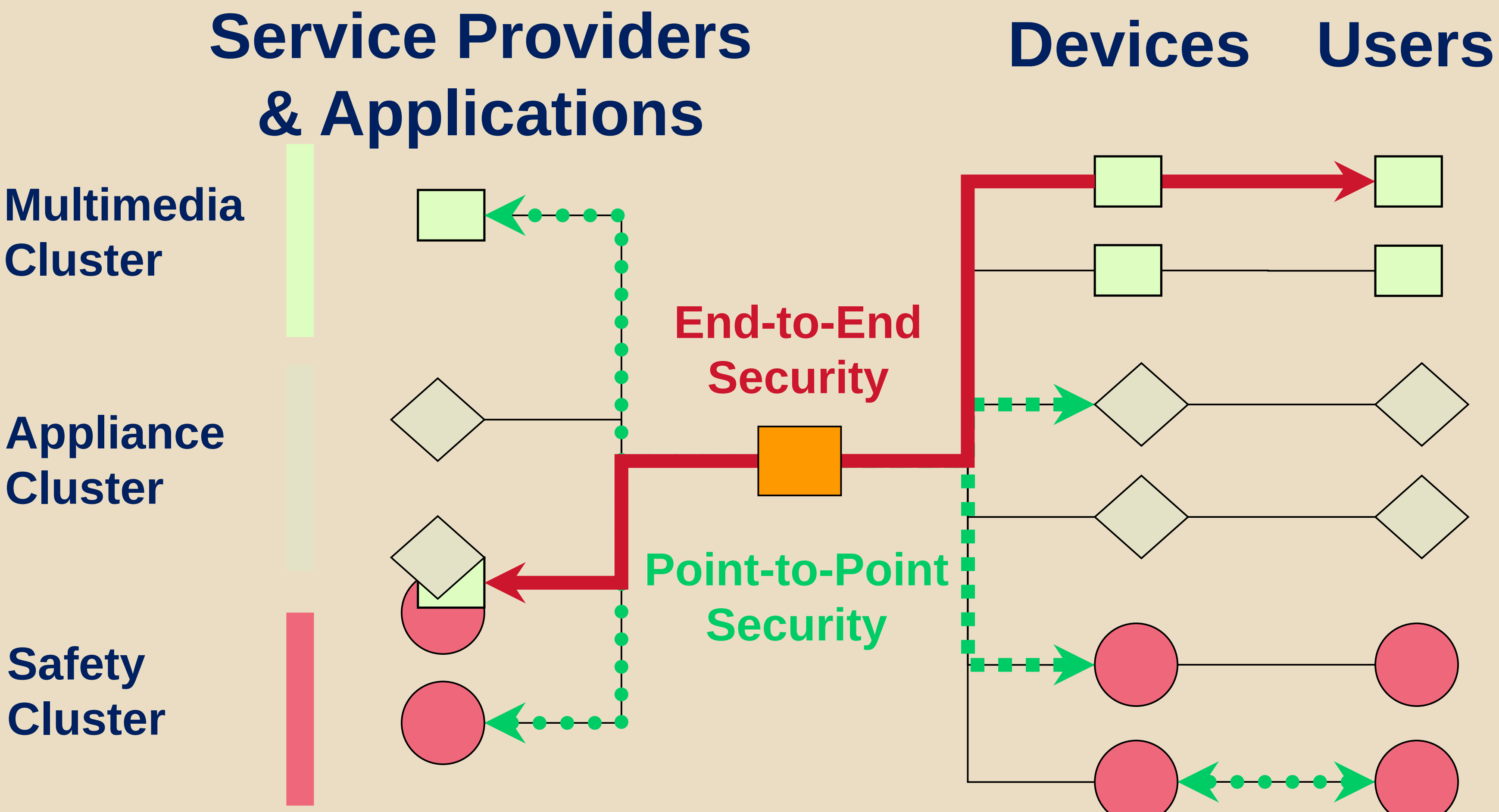
WHAT TO DO ABOUT IT?

- Large software vendors make lots of effort
  - Ongoing effort to improve security through its development process
    - Involves training and process improvements
  - Good practices:
    - Initial approach: freezing the current status
    - Only allow changes to improve overall security
- Good system design relies on embedded security
  - Simplifies security issues: no late add-on
  - Hides complexity of cryptographic protocols

GLOBAL SYSTEM OVERVIEW

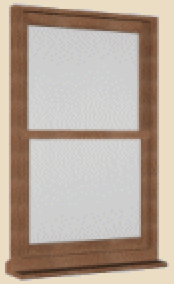


SECURITY VIEW



REAL LIFE THREAT – OPEN SESAME

- Third party’s benefit
  - Hacking/infecting remote control points
    - Very similar to botnet activities
  - Compromised meta-controller, e.g., IFTTT
    - Can provide full access to critical control points
    - Enables perfect burglary
      - Break-in & entry without signs of break-in!
  - Compromised device manufacturer’s control points
    - Alien firmware, Trojan behavior of *all* devices
- Self-benefit
  - Current state of the art allows fabrication of alibi ☺
    - Fake presence at home
    - Mimic normal behavior remotely



Disclaimer: not claiming the pictured items/service providers have been compromised already ☺
Images: http://www.sevenoaksart.co.uk

WHAT TO DO ABOUT IT? (DESIGN VIEW)

- Privacy by design
  - Avoid transporting and saving plaintext data to the cloud
    - Guarantee long-term security
  - Informed user consent & version control
  - Enforce information tagging
- Security by design – Adversary model?
  - Consistent deployment of a security vision saves time and money
  - Key material, set of trusted references: keys, certificates – TPM specifications
  - Enable decent user and system authentication & authorization
  - Consider use of tamper evident hardware where necessary – secure manufactory
- Manageability by design
  - Enable & use robust version and update control from the initial start
    - Firmware, operating system, application, application modules, device drivers
    - User data, configuration, consent
- Usability & configurabilty by design
  - Special focus on user friendliness & user/novice convenience

WHAT TO DO ABOUT IT? (DEVELOPER VIEW)

- What to focus on?
  - Applications/services
    - Long-term security & recovery from algorithm/key/security compromises
    - Consider algorithms and protocols as parameters
    - Validation of credentials & revocation
  - Network infrastructure
    - Device identification/authentication/authorization
    - Backend authentication/authorization
    - Denial of Service prevention & recovery
  - Devices have long lifetime
    - Cryptanalysis of algorithms
    - Side-channel analysis to retrieve long-term keys
    - Fault attacks, protocol poking

WHAT TO DO ABOUT IT? (DEVELOPER VIEW)

- Avoid reinventing the wheel
  - Get inspiration from Trusted Platform Modules, Digital Rights Management…
- Enable decent authentication & authorization
  - Devices, backend, users, services
  - Separate authentication from authentication
  - Network security protocols protect confidentiality and integrity
    - No protection of information authenticity out-of-the-box
- Centralize security knowledge in software/application architects
  - Implementers should not have to make delicate security decisions
- Good initial security design avoids hard to solve implementation issues
  - Goal: nearly-zero configuration
  - Security patches do not deal with inherent design flaws
  - Simple design is easily understandable/testable/auditable

WHAT TO DO ABOUT IT? (USER VIEW)

- Apply well known network segregation:
  - Demilitarized zones & self-controlled and managed security gateways!
- During configuration of intelligent devices
  - Prepare separate networks from normal network with Internet access
  - Use different settings to initialize/configure devices/services and to use devices/services
- After configuration
  - Disable Internet access of critical intelligent devices
    - Avoid burglaries (online & physical) ☺
  - Disable automated update functionality
    - Avoid unwanted/uncontrolled service disruption

GOOD PRACTICES

- Centralize security knowledge in software **architects** and application **designers**
  - Implementers should not have to make delicate security decisions
  - Cryptographic algorithms and protocols should be considered as modular building blocks
  - Consistent deployment of a security vision saves time and money
  - Security expertise concentrated in a few of the most trusted members of the development organization
  - Allows for better depth of knowledge
  - Results in more effective and secure results
- Good initial security design avoids hard to solve security issues
  - Security patches do not deal with inherent design flaws
  - Simple design is easily understandable/testable/auditable/updateable/upgradeable

ULTIMATE GOAL

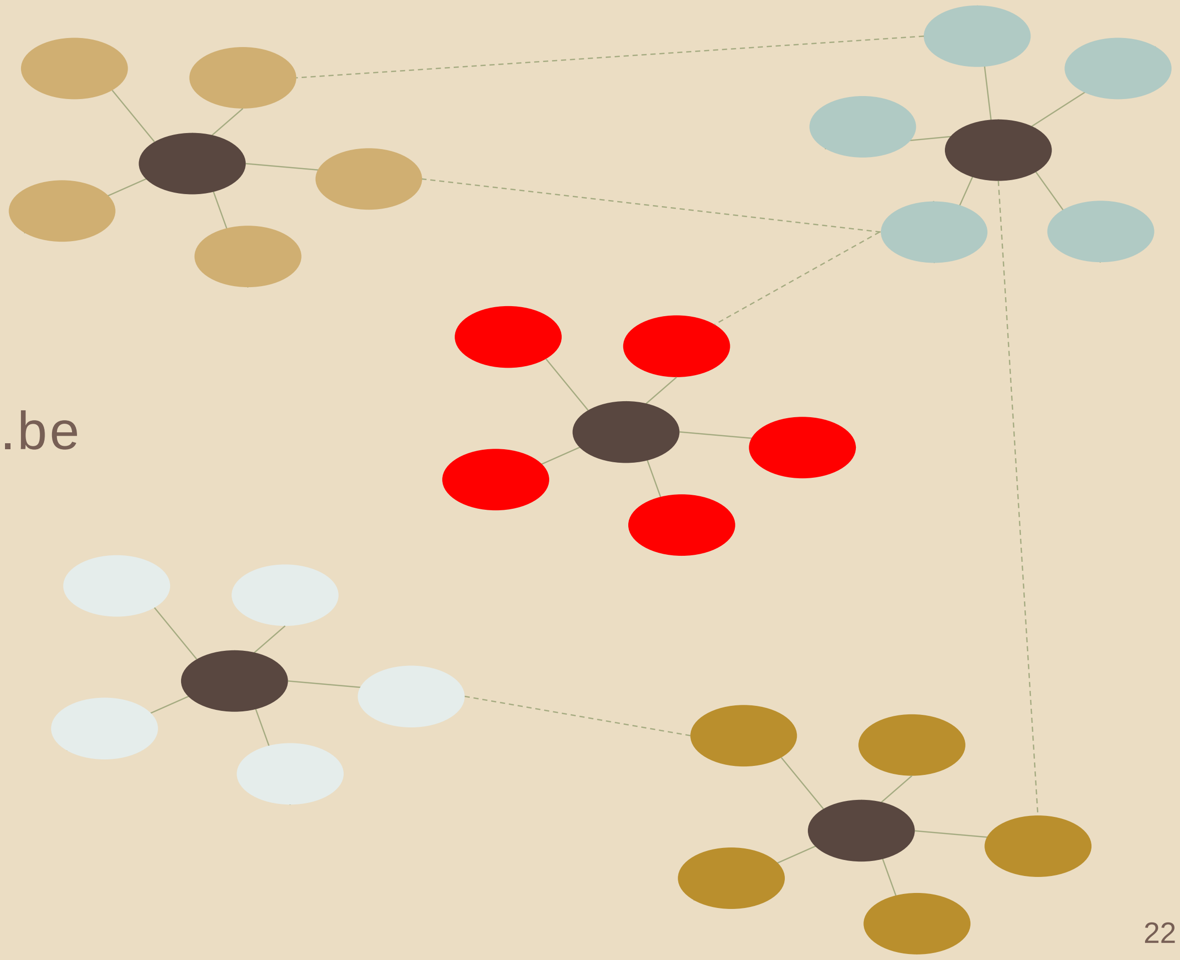
- Secure nearly zero-configuration
  - Simple hierarchy of devices, users, administrators, service providers
  - Seamless interoperability and interaction with other devices
  - Initialization of security parameters during device and service discovery
- Remote management of security parameters, software, configuration, users,…
  - ⇨Minimizes maintenance costs
  - ⇨Suited for a highly dynamic client-service architecture
- Simple and modular security mechanisms & system architecture
  - ⇨Ideal and easy to understand and verify

CLOSING REMARKS

- Use of Today's IoT devices provide
  - No privacy guarantees whatsoever
  - Fake belief you are in control
- About home automation
  - Not to be used for safety and security critical systems ☺

QUESTIONS?

- Contact details:
  - Email: Danny.DeCock@esat.kuleuven.be
  - Slides: http://godot.be/slides



PROTOCOL STACKS VIEW

User/Business Layer
Uses devices & services

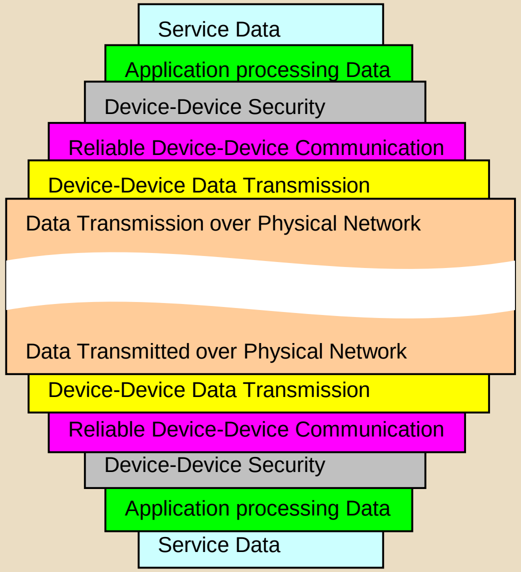
Application Layer (OSI Layer 7)
Offers Services to Users, Services and Devices

Security Layer (OSI Layer 5 – Session)
Protects Against Remote Evil Services and Devices

Transport Layer (OSI Layer 4)
Provides Reliable Communications

Network Layer (OSI Layer 3)
Provides Network Access

Data Link Layer (OSI Layer 2)
Communication Technologies, e.g., RF, WiFi, IR,…



LAYERED DEVICE VIEW