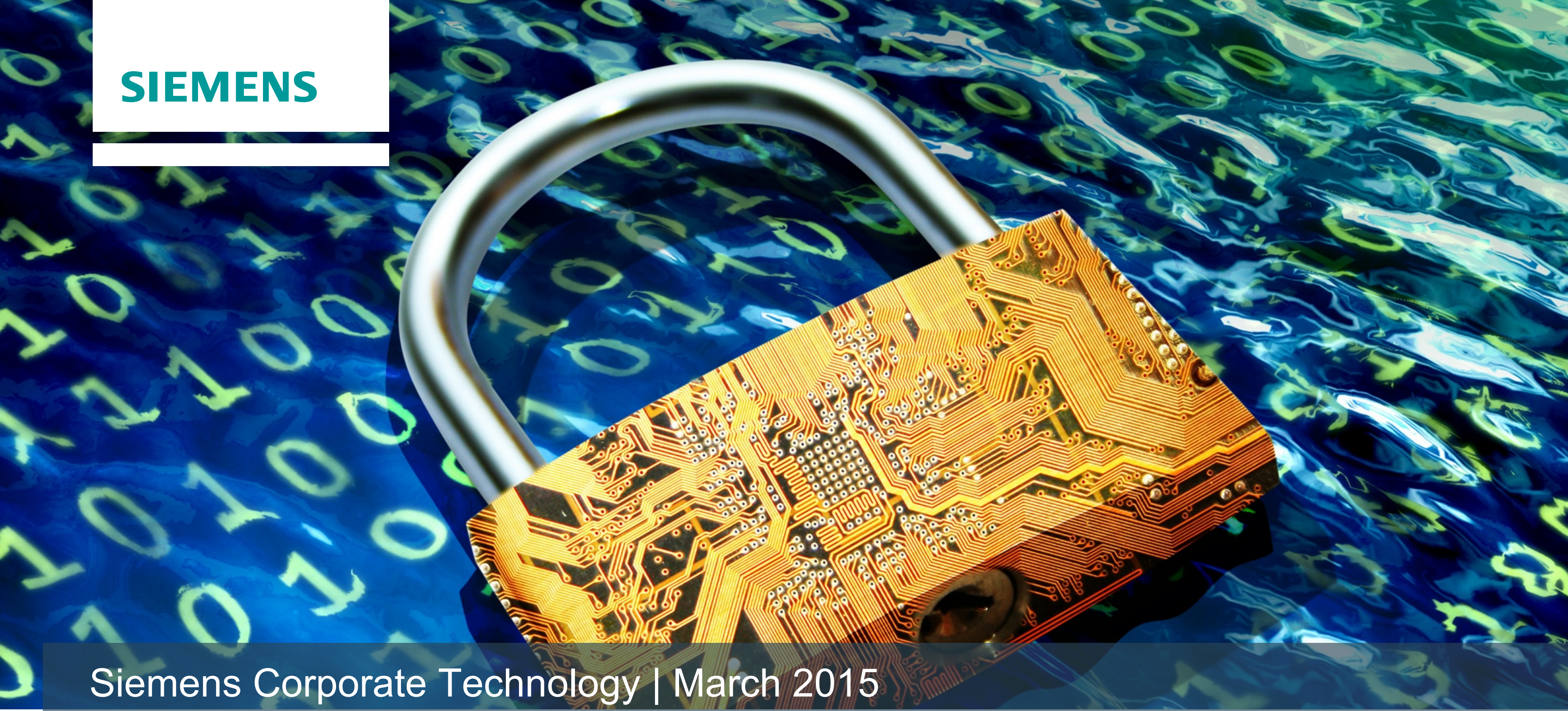
SIEMENS



Siemens Corporate Technology | March 2015

T2TRG Workshop: Research Questions for Security in IoT

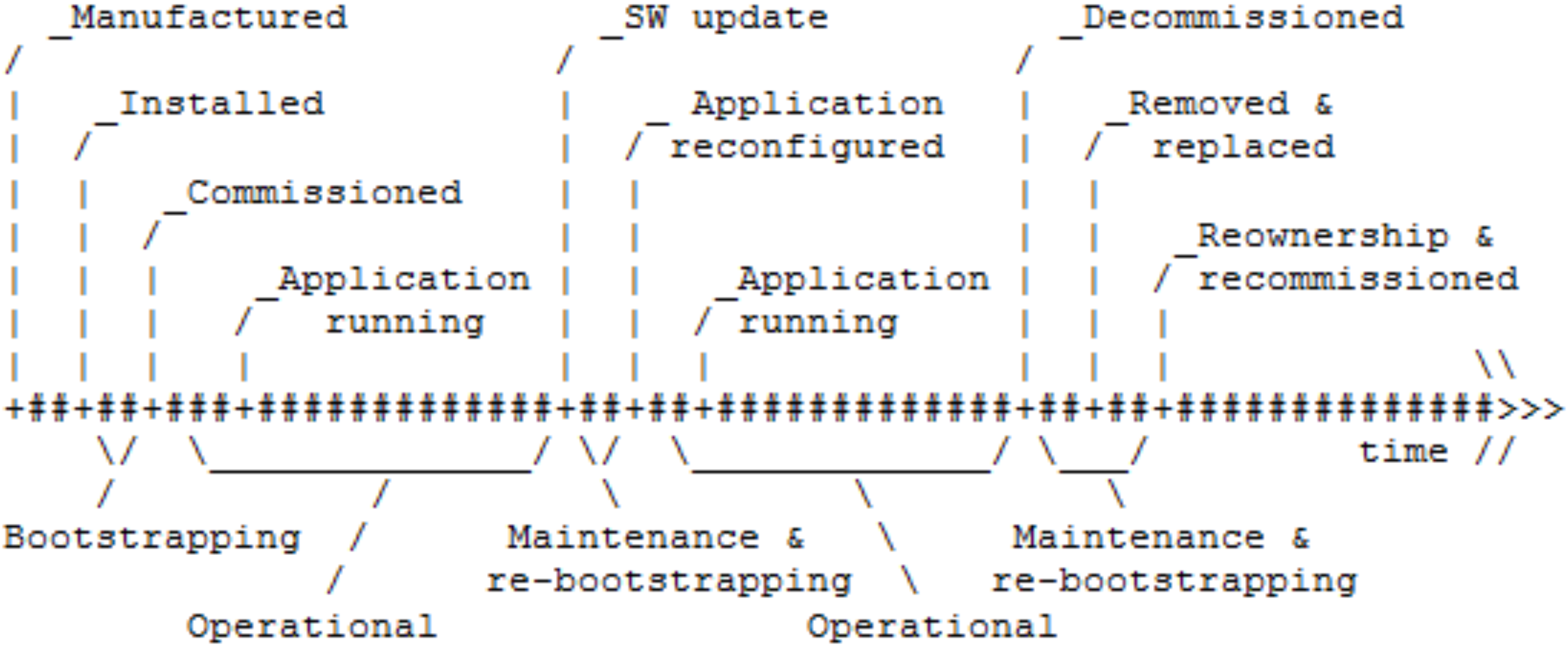
Objective: Problem Statement

- This presentation outlines IT-security related matters for T2TRG
- The slides
  - focus on width rather than depth
  - identify and explain starting points for consideration
  - address a general audience

Digital vs. Physical Goods

- The IT industry is used to processing digital goods. It has a ‘digital good’ mindset:
  - **Digital goods** — *reproduction, relocation of item instances at almost no cost*
  - Examples: Web pages, messages, contact/mapping information, mp3 files…
  - Aspects:
    - Static vs. dynamic objects
    - Human vs. machine-readable
- Things are physical:
  - **Physical goods** — *reproduction, relocation of item instances at cost*
  - Examples: lighting devices, smoke sensors, thermostats, controllers…
  - Aspects:
    - Consumer vs. investment goods
    - Individually vs. legal entity-owned

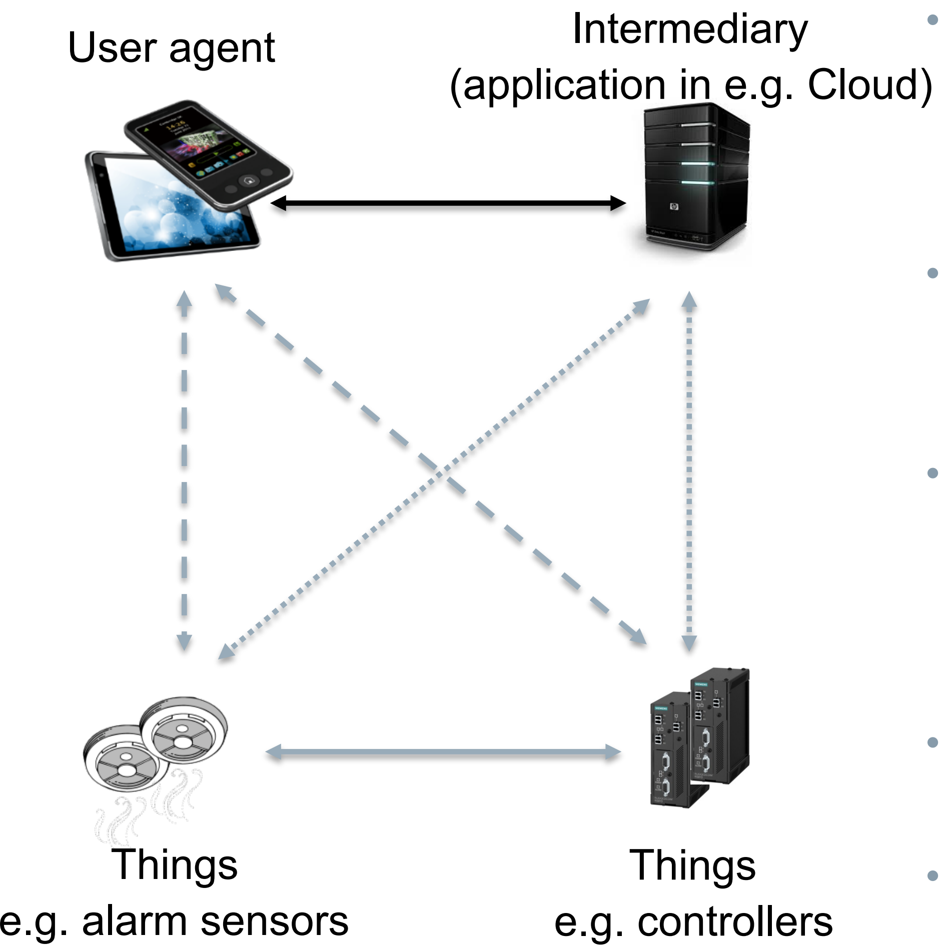
The Lifecycle of a Thing According to draft-garcia-core-security



```
  _Manufactured           _SW update          _Decommissioned
/                       /                   /
|   _Installed          |   _ Application   |   _Removed &
|  /                    |  / reconfigured   |  /  replaced
|  |   _Commissioned    |  |                |  |
|  |  /                 |  |                |  |   _Reownership &
|  |  |    _Application |  |   _Application |  |  / recommissioned
|  |  |   /   running   |  |  / running     |  |  |
|  |  |   |             |  |  |             |  |  |             \\
+##+##+###+#############+##+##+#############+##+##+##############>>>
    \/  \______________/ \/  \_____________/ \___/         time //
    /           /         \          \          \
Bootstrapping  /      Maintenance &   \     Maintenance &
              /      re-bootstrapping  \   re-bootstrapping
        Operational                Operational
```

[draft-garcia-core-security]

System Model



User agent

Intermediary
(application in e.g. Cloud)

Things
e.g. alarm sensors

Things
e.g. controllers

- *Allow things/devices to be engaged/engage*
- Variety of topologies
  - Direct interactions between things (T2T, U2T or T2U)
  - Mediated interactions (I2T, T2I)
- Variety of connectivity styles
  - Near field…wide-area
  - Intermittent…undisturbed
- Variety of communication patterns
  - Request/response
  - Publish/subscribe
  - One-way
- Variety of protocols
  - AMQP, CoAP, HTTP, MQTT, XMPP…
- Variety of constraints on things and networks
  - RFC 7228 classes 0/1/2

(User and) Thing Identity Model

- Repository perspective - input to entity authentication:
  - 1..n **identifiers** e.g. mail address or UUID value — serve the identification of the thing resp. a data object describing it
  - 0..n **credentials** e.g. password or shared secret key — serve the validation of claimed identity
  - 0..n **attributes** e.g. location or the RFC 7228 class of a thing
  - 0..n **affiliations** e.g. role assignment or group membership
- Security token perspective – output from entity authentication:
  - 0..n **identifiers** e.g. mail address or UUID value
  - 0..n **attributes** e.g. location or the RFC 7228 class of a thing
  - 0..n **affiliations** e.g. role assignment or group membership
  - 1..n **metadata** items e.g. security token issuer, authentication authority or method
- Notes:
  - Security token issuance is assumed to be specific for relying party components
  - Authorization may or may not use identifiers: identifier listings in rule sets, representation in security tokens is cumbersome in a case such as "*open door if from same location*"

Security Concerns

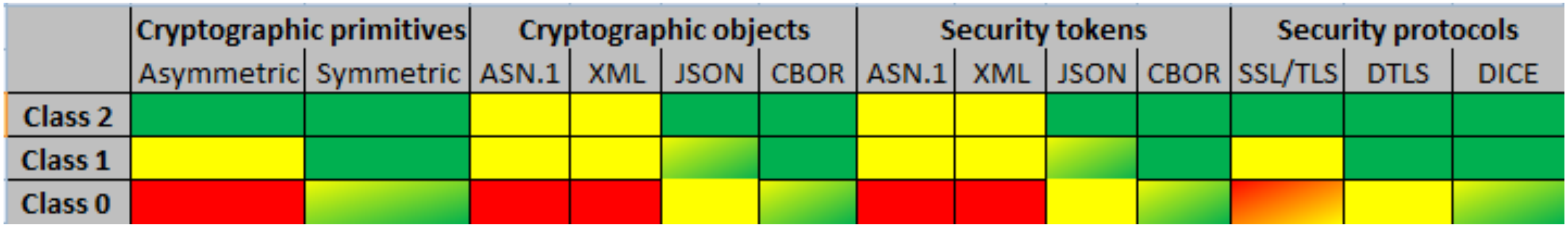
- **Entity authentication**
  - Rationale: *On the Internet nobody knows you are a dog.*
  - Authentication establishes confidence in the claimed identity of a caller (‘*I am John Doe, a member of the Super-Duper team*’) or callee (‘*I am bank.com*’)
- **Authorization**
  - Rationale: entity authentication (‘*This is John Doe*’) is necessary but not sufficient to determine what an entity should be allowed to do
  - Authorization serves to model, determine and enforce resource access rights
- **Secure communications** (encryption, message authentication)
  - Rationale: packets exchanges on IT-networks may be eavesdropped or manipulated
  - Encryption and message authentication safeguard messages through a hostile networks
- **Contextual security measures** (intrusion detection/prevention, throttling , risk-based authentication)
  - Rationale: increased connectivity and de-perimeterization demand new risk-management approaches
  - Means that react to presented messages/requests: intrusion detection/prevention, throttling and risk-based authentication allow to safeguard components that reside in a hostile environment

Means to Address these Security Concerns

- **Cryptographic primitives**: algorithms to transform data
  - Encryption vs. message authentication
  - Asymmetric (e.g. RSA, ECDSA) vs. symmetric (e.g. AES, SHA-2)
- **Cryptographic objects**: representations of transformed data along with metadata e.g. JOSE
  - Form factors: ASN.1 (e.g. PKCS), XML (e.g. XML Signature/Encryption), JSON (e.g. JOSE), CBOR (e.g. COSE)...
- **Security tokens**: formats to make assessments about system actors e.g. JWT
  - Form factors: ASN.1 (e.g. Kerberos tickets), XML (e.g. SAML assertions), JSON/CBOR (e.g. JWT)…
- **Security protocols**: means to exchange cryptographic objects or security tokens
  - Focus on exchanging cryptographically transformed data:
    - Transport-bound (protection of data-in-transit): SSL/TLS, DTLS, DICE
    - Information-bound (protection of data-at-rest): CMS, XML Signature/Encryption, JOSE/COSE — potentially enhanced by IoT/WoT-adequate freshness indicators
  - Focus on requesting/submitting security tokens:
    - Kerberos protocol (Kerberos ticket), SAML protocol (SAML assertions), OAuth (OAuth tokens, note: contents not defined), OIDC (JWT)

Their Fitness for Things

- The well-known Internet/Web security mechanisms do **not match** class 1/0 devices
- Results in a need to tune security mechanisms
- Required measures include:
  - **Down-scaling** of security system implementations
  - **Lightweight security** mechanisms covering
    - Cryptographic primitives
    - Cryptographic objects
    - Security tokens
    - Security protocols



	Cryptographic primitives		Cryptographic objects				Security tokens				Security protocols		
	Asymmetric	Symmetric	ASN.1	XML	JSON	CBOR	ASN.1	XML	JSON	CBOR	SSL/TLS	DTLS	DICE
Class 2	🟩	🟩	🟨	🟨	🟩	🟩	🟨	🟨	🟩	🟩	🟩	🟩	🟩
Class 1	🟨	🟩	🟨	🟨	🟨🟩	🟩	🟨	🟨	🟨🟩	🟩	🟨	🟩	🟩
Class 0	🟥	🟨🟩	🟥	🟥	🟨	🟨🟩	🟥	🟥	🟨	🟨🟩	🟥🟨	🟨	🟨🟩

Their Allocation in Things Lifecycle

- **Designing/developing**:
  - Choose cryptographic primitives and object formats, security token formats and security protocols
  - Create/supply implementations for the selected mechanism
- **Manufacturing/commissioning**:
  - Create a things instance (physical good) evtl. create/assign identity/keying information
  - Create configurations incl. deployment-specific identity/keying information
- **Operation**:
  - Employ the selected mechanism according to established configuration
- **Selling**:
  - Destroy operational configurations including deployment-specific identity/keying information
  - After change of owner: re-commission and re-use
- **Decommissioning**
  - Destroy operational configurations including deployment-specific identity/keying information

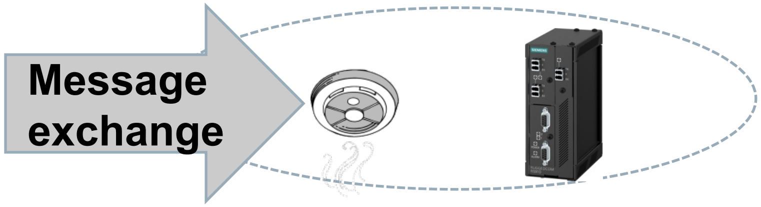
Entity Authentication — Callers

Message exchange

Things/devices as callers
(classes 0/1/2)

- The set of actors **increases by 1 order of magnitude** (approx. 7’’’ users, 50’’’+ devices). New actors have **new characteristics**:
  - Lack of user interfaces and displays
  - Unattended operation
  - Difficulties in keeping secrets secret
- The current user authentication practices rely on an **anti-pattern**:
  - Users or providers may leak credentials
  - Users forget credentials
  - Credentials get overexposed (HTTP Basic)
  - 3rd parties that ask users for shared secrets
- Results in a need to re-think mechanisms for the authentication of callers
- Required features include:
  - Device **identity bootstrapping**, **credentialing**
  - Device **authentication**

Entity Authentication — Callees



Things/devices as callees (classes 0/1/2)

- The current practices **do not match**
  - Kerberos: confined to Windows domains i.e. office/enterprise IT
  - SSL/TLS (PKI-based): see below
  - SSH (public key cryptography with no/ lightweight infrastructure): tailored according specific use cases in IT
- Results in a need to re-think mechanisms for the authentication of callees
- Required features: as for caller authentication
  - Device **identity bootstrapping**, **credentialing**
  - Device **authentication**

Public Key Cryptography/Infrastructure Checkpoints

SIEMENS

- Public key cryptography (asymmetric schemes without infrastructure or an ad-hoc key distribution model such as self-signed certificates):
  - **Complexity**: computational complexity (energy use, latency), code size (asymmetric algorithms) bearable for the intended things?
  - **Level of understanding**: sufficiently intuitive for the intended actors (designers, integrators, administrators, users)?
- Public key infrastructure: (supplementary means to distribute and manage public keys)
  - **Time**: do things have a perception of time and timeliness (notBefore, notAfter)?
  - **Naming**: does a naming concept for things exist and can it be mapped/adapted to the assumptions of the PKI technology
  - **Complexity**: code size (ASN.1) bearable for the intended things? Alternatively: offloading PKI object (certificates, CRLs, OSCP requests/responses) processing feasible with the intended things (added roundtrips)?
  - **Scale**: can the providers bear scaling to required object quantity?
    - Note: ca. 5’’ SSL/TLS server (leaf) certificates exist worldwide but 50’’’ devices projected to have Internet connectivity by 2020 — a factor of 10.000 for a technology (PKI) that is known to be somewhat tedious

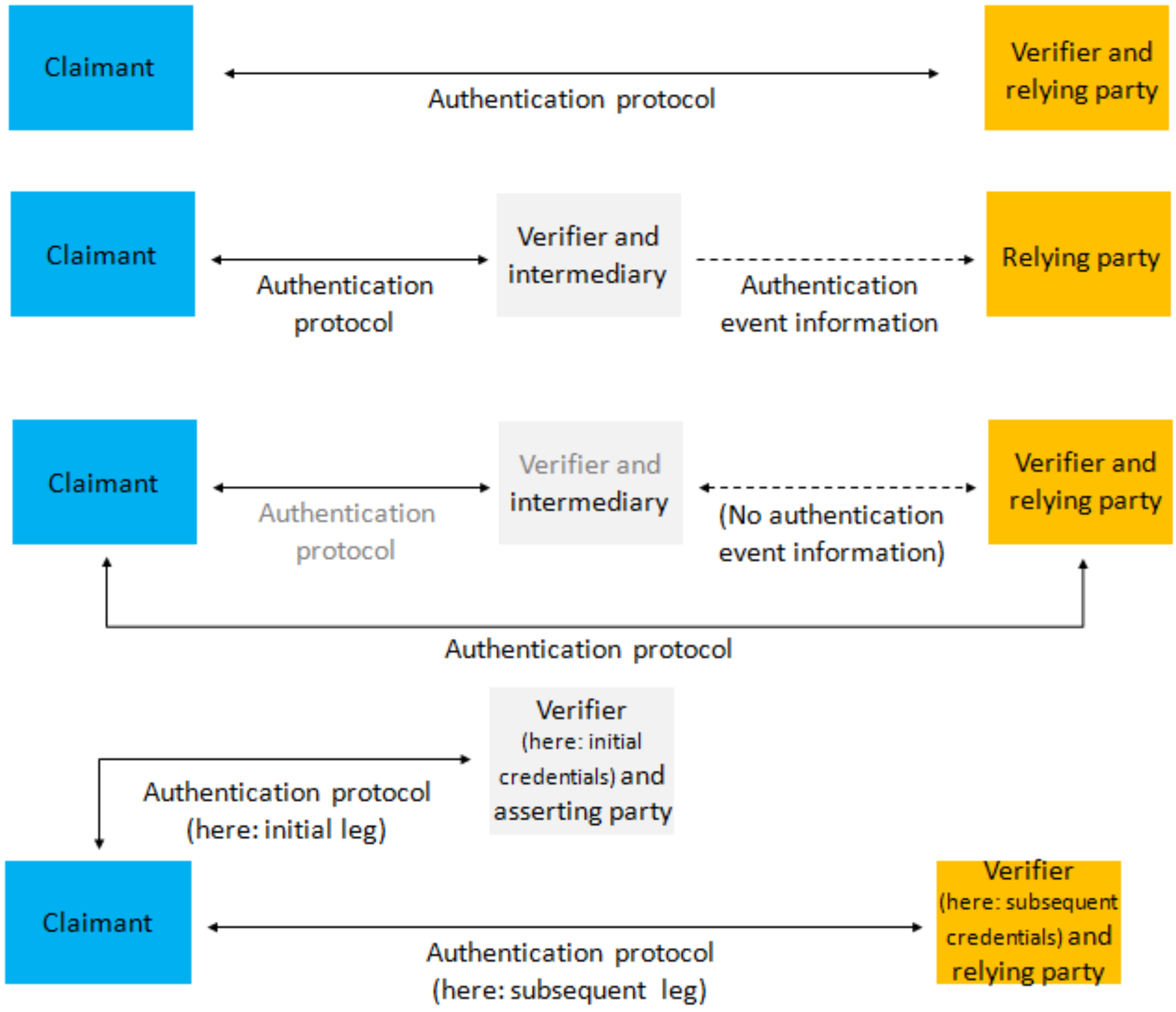
Entity Authentication – System Roles

- **Claimant**: entity that claims specific identity properties (e.g. "*I am John Doe*" and/or "*I live in Schenectady* ") and that needs to prove this identity claim
- **Asserting party** (aka **IdP**): component which authenticates entities and provides assertions about authentication events
- **Verifier**: component which challenges for user authentication and validates presented proof e.g. Web server or Web server extension module (Apache httpd module, Tomcat valve…)
- **Relying party**: component which trusts the verifier for establishing user authentication and takes dependent tasks e.g. Web application (servlets/JSPs/Jersey, PHP/Perl scripts…)

SIEMENS

Entity Authentication – Sample Patterns

- **Direct**:
- **Inline 3rd party, trusted**
- **Inline 3rd party, not trusted**
- **Online 3rd party**



Claimant

Authentication protocol

Verifier and relying party

Claimant

Authentication protocol

Verifier and intermediary

Authentication event information

Relying party

Claimant

Authentication protocol

Verifier and intermediary

(No authentication event information)

Verifier and relying party

Authentication protocol

Verifier (here: initial credentials) and asserting party

Authentication protocol (here: initial leg)

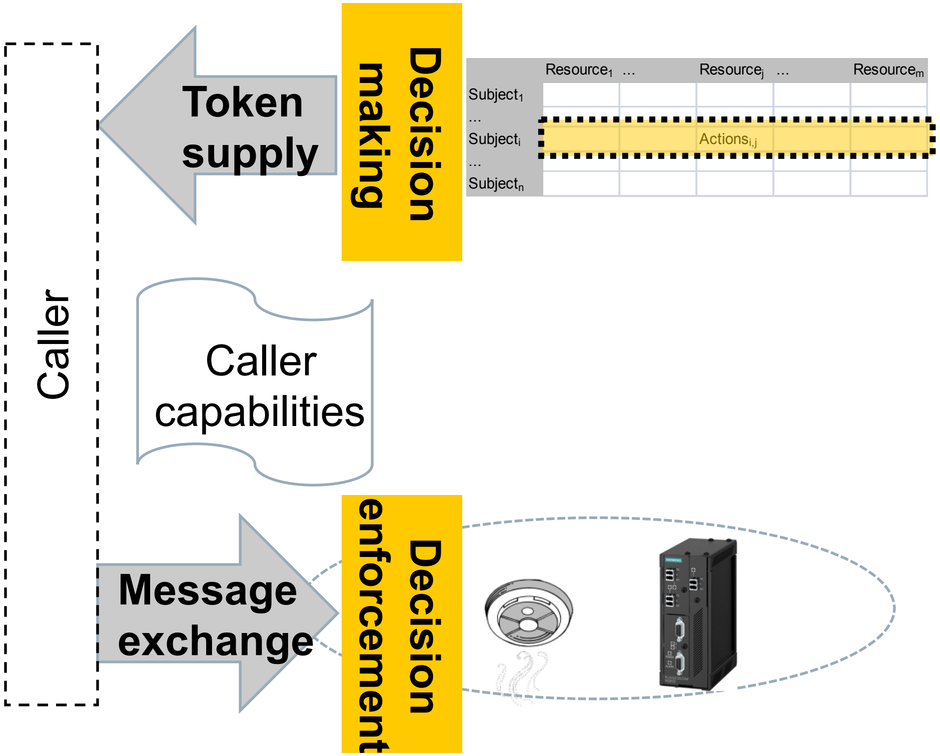
Claimant

Authentication protocol (here: subsequent leg)

Verifier (here: subsequent credentials) and relying party

Authorization

Things/devices as controllers (class 2)



	Resource$_1$	...	Resource$_j$	...	Resource$_m$
Subject$_1$					
...					
Subject$_i$			Actions$_{i,j}$		
...					
Subject$_n$					

Things/devices as callees (classes 0/1/2)

- **Decision enforcement** needs to happen **close to the resource**. It can typically not be offloaded from constrained things
- **Decision making** is a complex task (implements the access control matrix in some way) and needs to be **offloaded**
- Externalization of decision making prefers a **push** mode
  - Pull adds backchannel roundtrips per request
- This requires security tokens capable of describing **capabilities** of the requesting subject along with protocols to acquire, supply and possibly validate, revoke such objects
- These means have to be embedded with the protocol stack used to interact with the device
  - Corresponding means added to the HTTP stack (class 2+) in 2012 (OAuth 2.0)
  - Corresponding means for class 1/0 emerge just now

Authorization – System Roles

- **Decision enforcement**: component which obtains authorization decisions and enforces them
- **Decision making**: component or entity which obtains authorization decision requests, renders authorization decisions and responds with them
  - This can be a system component e.g. PDP that renders authorization decisions on base of an authorization rule-set
  - It can also be a human user that decides about instances authorization requests (the OAuth use case of O-to-O authorization e.g. *Do you allow printservices.com to access your jpegs stored at Google Drive*)
- **Policy making**: entity which establishes a rule-set for authorization decision rendering
  - This entity may be not be separate (see OAuth use case above)

Authorization – Decision Enforcement Allocation Options

- **Upstream from application** (e.g. reverse proxy or filter chain module): modules that monitors inbound (request) messages, checks authorization, lets authorized messages pass, responds with error messages else
  - Sufficient if authorization is unaware of application logic
- **Inside application**: modules that resides in the application code and that are called to check for authorization (various ways of defining the contract of such modules, implementing and integrating them)
  - Required if authorization is aware of application logic

Authorization – Decision Making Integration Options

- **Push into decision enforcement**: the decision enforcement component gets a request with information that expresses an authorization decision (or facilitates local decision making by simple means) and enforces it
- **Pull by decision enforcement**: the decision enforcement component gets a request without such information, transforms the access request into an authorization decision making request, passes it to the decision making component, gets the authorization decision response and enforces it
  - Note: this involves complex processing tasks; unlikely to match constrained things
    - Invoking external decision enforcement components adds expensive networks roundtrips (on a per-request or message base)
    - Invoking internal decision enforcement components increases code complexity (PDP component) as well as objects (authorization policies)

Authorization – Legal-Entity Owned Objects

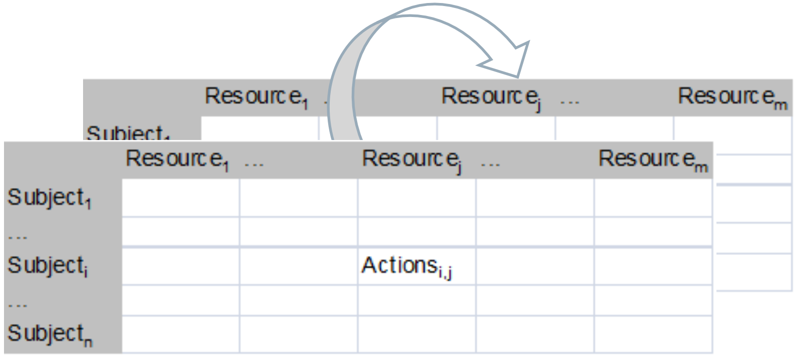
- Resource owner always represented by proxy persons (manager/admin…)
- Authorization system dynamics with a priori rules:

1. Create policy (manager/admin…)
2. Receive access request (application resp. its decision enforcement component)
3. Decide on access request (decision making component representing the legal entity)
4. Enforce decision (application resp. its decision enforcement component)

Authorization – Individually Owned Objects

- Resource owner usually not represented by proxy persons
- Authorization system dynamics with a posteriori rules (OAuth authz flow covering O-to-O authorization):
  1. Receive access request (application resp. its decision enforcement component)
  2. Decide on access request (individual)
  3. Enforce decision (application resp. its decision enforcement component)
  4. Store decision (application)
- Authorization system dynamics with a priori rules (UMA covering O-to-* authorization):
  1. Create policy (individual)
  2. Receive access request (application resp. its decision enforcement component)
  3. Decide on access request (decision making component representing the individual)
  4. Enforce decision (application resp. its decision enforcement component)

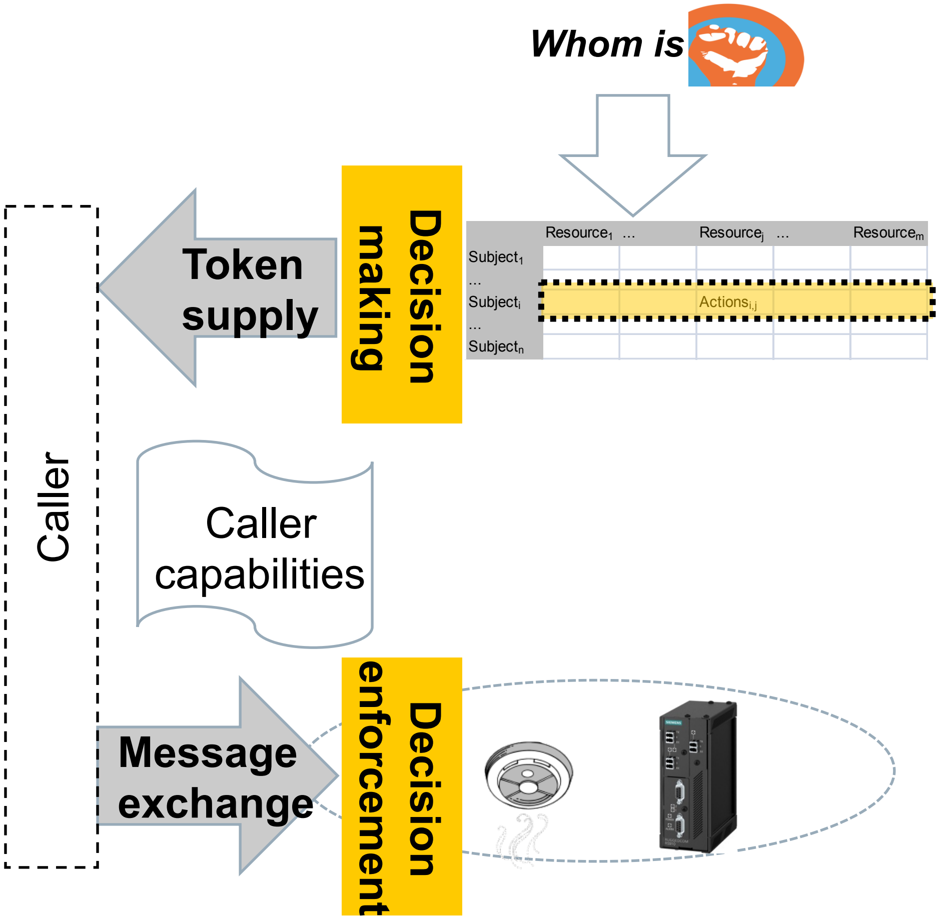
Authority of Authorization – Prior Art



	Resource$_1$	…	Resource$_j$	…	Resource$_m$
Subject$_1$					
…					
Subject$_i$			Actions$_{i,j}$		
…					
Subject$_n$					

- The **owner(s)** of an object are its root **authority of authorization**
  - This authority controls the contents of an access control matrix / its representation in implementation according provided tools
- Current practice is to understand and manage such authority in the case of digital goods
- Digital goods basics (reproduction and relocation at almost no cost) allow to address the management of ownership in a trivial way:
  - The resource owner is always known at digital goods creation time
  - Ownership of a digital item never gets transferred to another actor
  - Instead, objects are cloned (exploiting reproduction at almost no cost) and the new object is assigned to a new owner

Authority of Authorization – Things



	Resource_1	...	Resource_j	...	Resource_m
Subject_1					
...					
Subject_i			$\text{Actions}_{i,j}$		
...					
Subject_n					

- The current approaches **do not reflect the needs of physical goods.**
  - Change of ownership is commonplace in industrial IT. Sample scenarios:
    - Produce for an unknown customer, sell it
    - Produce for known customer who later sells it (and wants that to be possible without informing manufacturer)
  - The digital goods approach to reflect and manage ownership (clone the item) just does not do the trick for physical goods
- Support of this use case is mandatory. Its elaboration must address legal concepts:
  - Legal entity-owned goods: proxies (managers/admins…) are commonplace
  - Individually-owned goods: proxies an exception

Secure Communications

- Style of communication protection:
  - **Transport-bound security**: transport-level objects are cryptographically transformed
    - This protects data that is exchanged in the network
    - This protection is transparent to applications
  - **Information-bound security**: application-level objects are cryptographically transformed.
    - This protects the information content independent from its location
    - This protection is not transparent to applications
- Communication security services (mainline):
  - **Message encryption**
    - Transport-bound: SSL/TLS, DTLS, DICE – subject to negotiated cipher suite
    - Information-bound: PKCS#7, CMS (ASN.1), XML Encryption, JWE (JSON)
  - **Message authentication**
    - Transport-bound: SSL/TLS, DTLS, DICE – subject to negotiated cipher suite
    - Information-bound: PKCS#7, CMS (ASN.1), XML Signature, JWS (JSON)

Secure Communications (cont’d)

- Such mechanisms are needed if attackers have access to the raw bits representing exchanged information.
  - They also apply to communications between things: thing-friendly footprints of these options are needed esp. for RFC 7228 classes 0 and 1
- The selection of such mechanisms needs to consider
  - Both directed attacks and pervasive monitoring
  - Forward secrecy and its cost

Contextual Security Measures

- Contextual security services include:
  - **Intrusion detection/prevention**
    - Block suspicious traffic
  - **Throttling**
    - Enforce rate-limits, dynamically determine/adjust these limits
  - **Risk-based authentication**
    - Determine authentication schemes in a context-aware, adaptive way
    - Include step-up and re-authentication
- Forms of contextual security depends on the network exposure of things
  - In case of RFC 7228 class 0 things without own Internet connectivity, contextual security services need to be enforced at their controller or gateway
  - In case of RFC 7228 class 1/2 things with own Internet connectivity, contextual security services need to be enforced by either them or supplementary infrastructure components

Conclusions

- Security for the IoT/WoT presents a **challenge** for
  - Investment good vendors
  - Consumer good vendors
  - Web application and Cloud providers
- There will be **no one-size-fits-all** security solution for IoT/WoT
  - Constraints do vary too broadly across IoT/WoT scenarios
- Security for IoT/WoT is **no done thing:**
  - Innovations are needed e.g. means to reflect and manage device ownership
  - Further elaboration is needed e.g. means to manage device authorization as an end user

SIEMENS

Abbreviations

AMQP	Advanced Message Queuing Protocol
ASN.1	Abstract Syntax Notation 1
CBOR	Concise Binary Object Representation
CMS	Container-Managed Security
CoAP	Constrained Application Protocol
COSE	Constrained Object Signing and Encryption
CRL	Certificate Revoction List
DICE	DTLS In Constrained Environments
DTLS	Datagram TLS
HTTP	HyperText Transfer Protocol
I4.0	Industrie 4.0 (German term)
ID	IDentity
IdP	Identity Provider
IoT	Internet-of-Things
JOSE	Javascript Object Signature and Encryption
JSON	JavaScript Object Notation
JWT	JSON Web Token
MQTT	Message Queue Telemetry Transport
O-to-O	Owner-to-Owner
O-to-*	Owner-to-any
OAuth	Open Authorization
OCSP	Online Certificate Status Protocol
OIDC	OpenID Connect
PFS	Perfect Forward Secrecy
PKI	Public Key Infrastructure
SSH	Secure SHell
SSL	Secure Sockets Layer
T2TRG	Thing-to-Thing Research Group
TLS	Transport Layer Security
UMA	User-Managed Access
WoT	Web-of-Things
XMPP	eXtensible Messaging and Presence Protocol

Authors

Oliver Pfaff, oliver.pfaff@siemens.com, Siemens AG, CT RTC ITS

Presenter: Carsten Bormann cabo@tzi.org, Universität Bremen TZI