Modeling User-defined Trust Overlays for the IP-based Internet of Things

René Hummen, Christian Röller, Klaus Wehrle
Chair of Communication and Distributed Systems
RWTH Aachen University
Germany
{hummen, roeller, wehrle}@cs.rwth-aachen.de

I. Introduction

The advent of IP for constraint devices enables *individual* networked every-day devices, also called smart objects, to *directly* interact with a wide plethora of other devices and services. Specifically, IP technology allows for i) object-to-object communication across logical as well as technological network boundaries, and ii) object-to-Internet-service interaction and vice versa. For example, in a home automation scenario, a smart object controlling a radiator may get temperature information from a sensor in the same room and correlate this data to the weather information obtained from an Internet service. To protect smart objects from processing illegitimate or unwanted information, it is important that such communication is based on a model that reflects the trust relationships between networked entities. Based on such a model, smart objects can ensure the authenticity of trusted communication partners. Without the definition of trust relationships, the radiator controller in the example above may act on readings of a temperature sensor located in a neighbor's apartment or from a rogue Internet service. Moreover, correctly modeling and deploying trust models becomes essential when moving towards more security and privacy sensitive scenarios including networked alarm systems, medical systems, and intelligent industrial networks.

With DTLS [1], minimal IKE [2], and HIP DEX [3], protocols are currently under development at the IETF that aim at securing communication for smart objects at the IP layer and above. These protocols enable smart objects to mutually verify the identity of a communication partner. They assume the definition of trust relationships as a pre-requisite. However, defining and deploying the required trust models is challenging in many scenarios as smart objects often have a very limited interface for the interaction with the user.

In this paper we discuss the question: *"How can a smart object securely obtain information about trusted peers in a user-friendly way?"*. As a possible answer to this question, we introduce the entity of a trust point as the facilitator of trusted communication that maintains a model of trust relationships in the system. Users interact with the trust point to define trust relationships. The trust point then deploys trust policies on individual smart objects according to the generated trust model. To afford user-friendly, on-site configuration of smart objects, we also propose the use of smart phones for the user interaction with our system. The rich GUI of smart phones combined with their wireless network capabilities provide us with the missing user interface for smart objects without display or objects that are situated at inaccessible locations. As a large quantity of smart objects may be placed on-site, we additionally use the available sensors of a smart phone in order to provide a location-aware selection mechanism of smart objects.

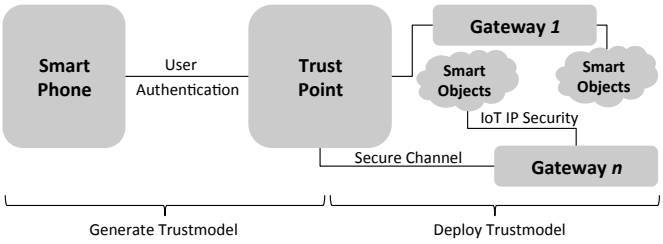

Fig. 1. Components of the proposed architecture.

II. Design of the Architecture

Our proposed architecture consists of four components: smart objects, gateways, a trust point, and smart phones (see Figure 1). *Smart objects* are resource constraint networked devices that run 6LowPAN and CoAP to communicate with each other and with Internet services. To prevent communication with illegitimate objects or services, smart objects only establish connections with peers for which local trust policies exist. Each smart object is connected to a *gateway* that is responsible for relaying packets addressed to devices located outside the local wireless network. Besides relaying IP packets, gateways also translate between different protocols (e.g., CoAP $\leftrightarrow$ HTTP) as required by the communicating peers. Gateways are interconnected via a common backbone network or the Internet. The *trust point* is the core entity in our architecture. It keeps track of the smart objects participating in the network and models the trust relationships between them. The trust point uses this model to derive trust policies for specific objects. It deploys these policies at the respective smart object via a secure connection that traverses through the object's gateway. Users connect to the trust point with their *smart phones* in order to set up or remove trust relationships between the smart objects of the network, thereby defining a

trust overlay with respect to the networked objects. The design of our architecture follows the typical interaction patterns and processes during the lifecycle of a smart object. Hence, we now outline these processes for our proposed architecture.

A. Object bootstrapping

For a smart object to participate in our system, it needs to become part of the trust model maintained by the trust point. This requires a two step process: i) an object registration phase and ii) a confirmation phase by the user. During the object registration phase, the smart object first establishes IP connectivity with the gateway. It then performs an authenticated handshake with the gateway to establish its cryptographic identity in the system. Candidate protocols for this purpose are DTLS or HIP DEX. If the handshake is successful, the gateway subsequently communicates the cryptographic tag of the object (i.e., the hash of the public key or of the DH public key) to the trust point. The trust point now has the necessary information to include the new object in its trust model. However, if the trust point directly allowed the inclusion of the object in its model, any object could become part of the trust overlay by performing a mere handshake. Hence, the object is still excluded from communication with other smart objects until its tag is confirmed by an authorized user.

During the confirmation phase, the user has to identify the partially bootstrapped object. We envision the use of QR codes, which the user scans with his smart phone during the confirmation phase. The QR code encodes the object's cryptographic tag and may, e.g., be placed on the packaging of the object [4]. To afford subsequent *location-based* on-site trust configuration for smart objects, the user additionally determines the object's location (e.g., based on GPS or WiFi fingerprinting). Finally, the user sends this information to the trust point, where it is stored and correlated to the known object tags. If the trust point finds a matching tag, it allows the inclusion of the smart object in the trust model.

B. Object pairing

To enable trusted communication between smart objects, the user needs to insert a corresponding trust relationship in the model maintained by the trust point. To this end, the user subsequently selects two objects during a pairing phase and instructs the trust point to pair the two objects in its trust model. The trust point then deploys new trust policies at the involved objects by sending them the cryptographic tag of each other. On receipt of the cryptographic tag from the trust point, a smart object stores the tag in its list of authorized communication partners. The paired objects may then perform the preferred candidate protocol for the cryptographic handshake at a later point in time in order to identify each other as trusted peers and communicate securely.

As location information of the smart objects is available at the trust point, the user may take advantage of this information when selecting objects on-site. To this end, he uses the smart phone to determine his location with the same mechanism used during the bootstrapping process. The location-aware selection mechanism enables the trust point to search its trust model for objects in the vicinity of the user and returns these as candidate selections to the user. Furthermore, as the cryptographic identity of an object is of little meaning to the user, the trust point may provide additional meta information about the object to the user (e.g, a device name or a description). This enables the user to uniquely identify an object in case of multiple candidate selections.

C. Object mobility

Our architecture supports mobility of smart objects in a natural way. Mobile objects may leave the range of their current gateway and come in range of another gateway. The object then performs the object registration phase of the bootstrapping process and authenticates itself with the new gateway. When the gateway sends the object tag to the trust point, the trust point recognizes the mobile object as an already bootstrapped participant. As the object tag only contains cryptographic, location-independent information, neither the trust model at the trust point nor the local trust policies at the smart objects need to be adapted. However, the location-aware selection mechanism would fail for mobile objects because they do not have a fixed location. Hence, mobile objects inherit the location of the gateway they are currently connected to.

III. Current State of the Prototype

The early prototype of our proposed architecture is based on commodity hardware and open source software. Currently, the gateway and trust point functionality is combined in a single hardware platform. Here, we use the router hardware Linksys WRT160nl with the open source Linux-based operating system OpenWRT. As smart objects, we use Tmote Sky motes running Contiki 2.5. We use the available 6LoWPAN stack for IP connectivity and CoAP for communication at the application layer. Smart phones are based on the open source operating system Android. Our prototype currently supports basic bootstrapping and configuration of objects located outdoors.

IV. Conclusion

In this paper, we identify the need to model trust relationships in order to secure communication in the IP-based Internet of Things. We briefly discuss an architecture that allows users to define trust overlays with respect to objects participating in the network. Furthermore, we show how smart phones can be used in our architecture to provide a user-friendly, location-aware mechanism to set up trust relationships between smart object. The such established trust relationships can then be used in security protocols such as DTLS and HIP DEX that are currently under development at the IETF.

References

[1] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security," RFC 4347 (Standards Track), IETF, 2006.

[2] T. Kivinen, "Minimal IKEv2," Internet-Draft (work in progress), draft-kivinen-ipsecme-ikev2-minimal-00, IETF, 2011.

[3] R. Moskowitz, "HIP Diet EXchange (DEX)," Internet-Draft (work in progress), draft-moskowitz-hip-rg-dex-05, IETF, 2011.

[4] J. Arkko and A. Keranen, "CoAP Security Architecture," Internet-Draft (work in progress), draft-arkko-core-security-arch-00, IETF, 2011.