Security Bootstrapping for Networked Devices

Christian Rohner, Uppsala University

Abstract: In this short paper I present two mechanisms to bootstrap security relations between networked devices: the ownership model and a security policy. The ownership model assures security relations between devices owned by the same user, and the security policy defines security relations to other devices, assigns rights to relations, and supports authentic key exchange.

1 Introduction

We observe the trend that more and more devices get equipped with some computing power and a communication interface. The communication interface is typically a short-range wireless technology that allows the devices to interact whenever and wherever they meet without relying on central infrastructure. Such networked devices often store personal and private information, have limited resources, and operate in an open and unknown environment. It is therefore important to restrict access to information and resources. The necessary security mechanisms (i.e., access control and encryption if necessary) are well known and present no new challenges. However, all these mechanisms rely on security relations such as shared cryptographic keys or certificates authenticating keys. It is not clear how to setup security relations in an environment of networked devices without requiring the user to be an expert and without relying on central infrastructure.

The following Section 1 introduces to the problem of building security relations between devices. In Section 3 and Section 4 I present two contributions to build security relations, the ownership model and the security policy, respectively. In Section 5 I summarize the work and provide an outlook to future research.

2 Building Security Relations

A security relation between two devices can be a common shared cryptographic key, or an authentic copy of a public key. It allows the two entities to exchange messages in an authentic and/or confidential way. However, security relations need to be bootstrapped, that is, keys have to be generated, distributed, and authenticated. These operations cannot be expected to be done by an average user, in particular not on devices with a limited user interface.

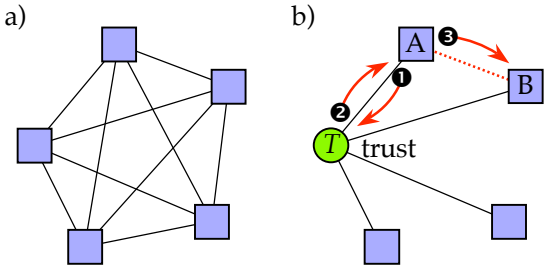

Figure 1: Two scenarios to bootstrap security relations: a) using predefined relations, or b) using a trusted entity.

2.1 Challenges

Because it is not realistic and necessary to predefine security relations between all devices (see Figure 1a), other alternatives have to be found. The common way is to use trust to reduce the number of initially required security relations (see Figure 1b): A trusted entity T can, for example, create a bilateral key for two devices A and B if both, A and B have a bilateral key with T and trust T to generate random keys and to forward only authentic information (see Figure 1b). Unfortunately, it is not practical to find one single entity T that is trusted by all other entities. Using a web of trusted entities would solve at least this concern, but is a delicate process because two devices that want to

build a relation must find a path through the web on which they trust every single entity.

The problem to be solved is to find a bootstrapping process to build security relations between devices that want to interact with one another without relying on predefined relations, central services (e.g., an administrator), or the availability of dedicated entities such as a trusted third party.

2.2 The Resurrecting Duckling Policy Model

Ross Anderson and Frank Stajano were the first that recognized the importance for security relations between networked devices. In their Resurrecting Duckling Policy Model [1, 2], they propose two basic elements:

- *Secure Transient Association*: exchange of a shared secret during physical contact (pairing) representing a master–slave relation between the devices.
- *Default policy*: the master device can access all services of the slave device; no other device is allowed to use services. One of the services accepts policy updates.

While the relation between devices is a static master–slave relation in the original paper [1], an extension to peer-to-peer relations is presented in [2] by describing relations to other devices in the security policy.

The pair-wise master–slave relations between devices introduce dependencies between devices that limit the usability and are prone to loss of devices. Although peer-to-peer relations partially address this shortcoming, the model does not suggest how the credentials (i.e., keys and certificates) to represent these relations could be described in the policy in an authentic way. Further, the lifecycle of a security association is rather static: Delegation and exception handling (i.e., loss of devices) are not supported.

2.3 Contributions

In the next two sections I present an extension to the Resurrecting Duckling Policy Model that addresses these shortcomings. The *ownership model* builds real peer-to-peer security relations between all devices owned by the same user and thereby strictly defines what devices that are trusted. The *security policy* defines relations to other devices, assigns rights to security relations, and supports authentic key exchange.

3 Ownership Model

My approach to build relations between devices uses a (random) cryptographic key pair as identifier of a device. When two devices from the same user get paired, one device creates a certificate for the other device by signing the identifier (i.e., the public device key) of the other device. The resulting certificate chain leads towards one of the user's devices and is used to recognize other devices owned by the same user (i.e., siblings). The default security policy of a device defines the same rights as the Resurrecting Duckling Policy Model.

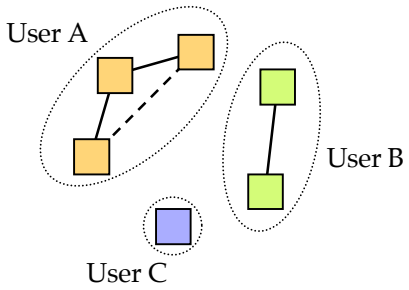

Figure 2: The ownership model builds security relations between devices owned by the same user.
Legend: *solid line* - security relation represented by a certificate. *dashed line* - security relation implied by a certificate chain.

The advantage of this approach is that a new device has to be paired only with one device to build up authentic relations to all other devices of that user. These relations provides redundancy to cope in situations where devices get lost or delegated, because a device is not dependent on a dedicated other device such as in master–slave models.

4 Security Policy

Every device has its own security policy to make access control decisions. The proposed security policy expressed in the Security Policy Definition Language (SPDL, [3]) is designed to not only describe access rights, but also allow remote configuration and authentic key exchange to support devices with a limited user interface and to introduce security relations to other devices, respectively.

4.1 Letter of Authority

Device security policies expressed in SPDL are a sequence of policy updates applied to the default policy. The letter of authority is a signed policy update that allows authentic remote configuration

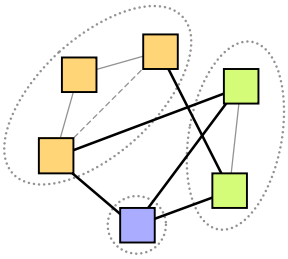

Figure 3: The security policy defines security relations between devices owned by different users.

of the device policy. It has two useful properties. First, the letter of authority allows to transport a key from one device to another in an authentic way, and second, allows to pass-on a policy update to the target device via other devices in case there is no direct connectivity.

4.2 Authentic Key Exchange

SPDL describes devices with credentials such as keys (i.e., the device identifier) or certificates for expressing roles that a device assumes. Instead of configuring these credentials in the policy, SPDL allows to set a wildcard specifying the conditions to accept the credential of the next device that gets paired with the target device. The involved devices thus exchange their credentials themselves without requiring the user to cope with cryptographic material.

5 Towards Self-Configuration: Conclusion and Ongoing Work

The presented work is part of the authors PhD thesis [3]. This short-paper summarizes mechanisms for building security and trust relations between devices, to assign rights to these relations, and to do authenticated key exchange. Details and a validation of the presented mechanisms can be found in [3], a proof of concept was done with a prototype implementation. Other work covered in [3] and ongoing research includes:

- *Delegation of Ownership*: During its lifetime, a device might change its owner, or the owner wants to hand over a device to another user. The provided mechanisms support such delegation with the possibility to restrict the functionality of a device, and to take back ownership.
- *Exception Handling*: Devices can be lost or stolen. Recovery mechanisms make use of the redundant security relations among devices owned by the same user to propagate recovery lists and re-assign certificates.
- *Limited Devices*: Networked devices often have limited resources that do not allow to implement the specified protocols. In particular cryptographic public-key operations are a big hurdle for limited devices. The architecture integrates devices with limited resources by means of a security proxy.
- *Privacy*: Privacy gets increasingly important in the context of networked devices because personal and private information and resources are exposed to an open and unknown environment. Besides of the information stored on the devices itself, also metainformation such as the users identity and location needs to be protected. My work concentrates on the use of pseudonyms and secure service discovery protocols to provide the presence of devices only on a need-to-know basis.

Most of the devices that we intend to connect will be personal and private devices of our daily life. Security aspects are therefore important. However, the most important aspect is the ease-of-use because the user cannot be expected to configure anything. However, I consider the security bootstrapping as a first step towards the ambitious goal of *self-configuration* in dynamic networks.

References

[1] F. Stajano and R. Anderson. The Resurrecting Duckling: Security Issues in Ad-Hoc Wireless Networks. In M. Roe B. Christianson, B. Crispo, editor, *Security Protocols, 7th International Workshop Proceedings*, Lecture Notes in Computer Science. Springer-Verlag, 1999.

[2] F. Stajano. The Resurrecting Duckling - what next? In M. Roe B. Christianson, B. Crispo, editor, *Security Protocols, 8th International Workshop Proceedings*, Lecture Notes in Computer Science. Springer-Verlag, 2000.

[3] C. Rohner. *Security in Ad-hoc Distributed Systems*. PhD thesis, ETH Zürich, 2003.