Application for a Simple Device Authentication Framework: Device Authentication Middleware using Novel Smart Card Software

Manabu Hirano¹, Takeshi Okuda², Suguru Yamaguchi²

¹ Toyota National College of Technology, Toyota, Aichi 471-0565, Japan
² Nara Institute of Science and Technology, Ikoma, Nara 630-0192, Japan
hirano@toyota-ct.ac.jp, {okuda, suguru}@is.naist.jp

Abstract

Our proposed simple device authentication framework [1] aims to provide device-oriented authentication and authorization mechanisms for non-PC Internet-ready information appliances. The purpose of the framework is to prevent device spoofing, and to restrict unauthorized access to the device in a future ubiquitous network. We have developed the novel smart card software to achieve peer to peer device-oriented PKI authentication and authorization on the secure tamper-resistant smart card. We assume that the smart card will be attached to a device such as an information appliance. In this paper, we show the prototype implementation of authentication middleware using our novel smart card software and the IKEv1 program. The IKEv1 program cooperates with our novel smart card software to achieve peer to peer production-level identity authentication for devices. Thus, our framework and its application shown in this paper will help to realize secure ubiquitous computing for Internet-ready home appliances.

1. Introduction

Future ubiquitous networks will be connected to a large number of non-PC Internet-ready information appliances. We have proposed an authentication framework not for human beings but for devices [1]. In our proposal, a home appliance attached with our proposed smart card can call our device-oriented authentication and authorization APIs from the smart card. Our proposed smart card can register a device’s identity for device-oriented authentication, and an owner’s attributes for access control for the devices.

A non-PC device such as a home appliance generally has only a minimum human interface, unlike a traditional PC with a full keyboard and display, making it difficult to input complex authentication data. Our proposal for novel smart card software with special configuration tools [1] will enable users to set up device-oriented authentication data, despite a poor user interface, by separating device-oriented authentication and authorization functions from the device itself. Consequently, the accessed device is able to call device-oriented authentication and authorization APIs from the smart card. As a result, the smart card that is attached to Internet-ready appliances can realize a peer-to-peer PKI (Public Key Infrastructure) authentication and authorization mechanism without an extensive interface. Fig.1 shows an example of device-oriented authentication in a ubiquitous network.
2. Overview of simple device authentication framework

Fig.2 shows the use case diagram for proposed framework. Our novel smart card software for the simple device authentication framework provides two functions described in section 2.1., 2.2.

2.1. Production-level identity authentication using public key certificate

During the production phase, our framework allows manufacturers to register a production-level identity such as manufacturer’s unique ID and device’s serial number into a smart card. To achieve production-level authentication mechanism, we employ the device’s X.509 Public Key Certificate (PKC) [2] and its attribute fields to store its public key and production-level identities. The product is shipped with this initialized smart card. As a result, the initialized smart card can handle the device’s production-level identity. We have developed an initialization tool for manufacturers to set up a production-level identity for the smart card [1]. In section 3, we will show the prototype implementation of the production-level identity authentication middleware using the IKEv1 protocol.

2.2. Owner-level attributes authorization using attribute certificate

After purchasing the product, our framework enables users to register owner-level attributes (distinguishable unique names assigned by the owner) on to their smart card. To achieve owner-level attributes authorization mechanism, we employ an Attribute Certificate (AC) [3] to store owner-level attributes, and an Access Control List (ACL) to restrict the access using the peer connecting device’s attribute. As a result, a user can set up her or his own device to restrict access from other devices. We have developed a personalization tool for the device owner to set up owner-level attributes to the smart card [1]. We assume that authorization functions in the smart card are called by the device-controlling application.

2.3. Implementation for novel smart card software

Fig.3 shows the design of our proposed novel smart card software. We have implemented this software running on Axalto Cyberflex e-gate 32k smart card [1]. Our software employs Java Card API 2.1.1 [4]. Table 1 shows the smart card specification of our implementation. Our novel smart card software provides (1) Authentication module using a private key and PKC, (2) Authorization module using AC and ACL, and (3) IPsec configuration provider to configure IP security policy. We have implemented these functions as a Java program running on the smart card. Communication between the smart card and the middleware is processed using the ISO-7816 standard [5].

3. Design of peer to peer device-oriented authentication middleware using IKEv1

In this paper, we have implemented a peer to peer device-oriented authentication middleware as an application of a proposed simple device authentication framework. The two devices which are trying to connect each other make it possible to execute strong PKI authentication using the middleware to cooperate with the smart card. We employ IKEv1 (Internet Key Exchange, version 1) [6] protocol to execute mutual authentication between the two devices. We also employ IPsec [7] protocol to encrypt communication between the two devices.
Our peer to peer device-oriented authentication middleware is executed on a Linux box. We assume that the Linux box is attached to an information appliance by RS-232, USB or Ethernet, and so on. The Linux box controls the information appliance. (ex. Recording a TV program on a DVD recorder) The Linux box has to equip the smart card reader. In this paper, we utilize the smart card as a small USB device shown in Fig.4. Our developed middleware running on a Linux box can call device-oriented authentication API from the smart card. The smart card software executes authentication functions and returns the result only. Consequently, the Linux box can authenticate the peer device’s production-level identity described in section 2.1.

Fig.5 shows our proposed peer to peer device-oriented authentication middleware system running on the Linux box. At the boot time of the Linux box, the IPsec policy management program calls the IPsec configuration provider from the smart card. ((3) in Fig.3 and Fig.5) It sends an IP security policy file to the Linux box. Thus, the IPsec policy management program sets up the IP security policy to the SPD (Security Policy Database) [8] in the kernel.

Fig.6 shows the IKEv1 negotiation for the main mode with signature authentication. IKEv1 has 2 phases to establish IPsec-SA [6]. IKEv1’s main mode is the “Phase 1” exchange. The authentication module in the smart card generates a signature (SIG_I for initiator, SIG_R for responder). Furthermore, the authentication module sends the PKC (CERT_I, CERT_R) related to the production-level identity (manufacturer’s ID and product serial number) from the smart card to the Linux box. Thus, the IKEv1 program can send them to the peer IKEv1 program. Consequently, the Linux box can authenticate production-level identity. Finally, the IKEv1 program establishes IPsec-SA and stores it to the SAD (Security Association Database) in the kernel.

4. Implementation of peer to peer device-oriented authentication middleware

Table 2 shows the software used with the implementation described in section 3. USAGI [9] is for the implementation of IPv6/IPsec stack in Linux. Our implementation of the proposed system employs an USAGI stack. Racoon [10] is the proven implementation of the IKEv1 protocol. We currently employ IKEv1 protocol. To employ the new IKEv2 protocol [11], we have to change an IKEv2 program to call the authentication module from the smart card. However, the modification of the IKEv2 program is simple, because it just replaces the sign functions with the smart card functions. Our implementation employs RSA 1024 bit key and SHA-1 hash algorithm. We utilize a setkey program [10] as an IPsec policy management program to add, update, dump, or flush SAD and SPD entries in the Linux kernel. Our implementation extends the setkey to load the IP security policy from the smart card. This process only executes at the boot time of the Linux box.

5. Performance measurement

In this section, we show the result of the performance measurement. Table 3 shows the hardware specification.
Table 3. Hardware specification of the Linux box

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Pentium M Processor 1.6 GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>512 MB</td>
</tr>
<tr>
<td>Network</td>
<td>100 Base-TX</td>
</tr>
</tbody>
</table>

Table 4. Process time of the IKEv1 program

<table>
<thead>
<tr>
<th></th>
<th>time [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load X.509 Certificate (942 byte) from the card</td>
<td>8.477</td>
</tr>
<tr>
<td>Create signature of IKE phase 1 payload in the card</td>
<td>5.977</td>
</tr>
</tbody>
</table>

Table 5. Comparison of the process time of IKE phase 1 exchange

<table>
<thead>
<tr>
<th></th>
<th>time [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional IKEv1 program</td>
<td>0.675</td>
</tr>
<tr>
<td>Proposed IKEv1 program</td>
<td>12.132</td>
</tr>
</tbody>
</table>

of the Linux box. Table 4 shows the process time of the modified IKEv1 program. Table 5 shows the comparison of the proposed system with the conventional system. The process time of the phase 1 exchange is 12.132 sec in the proposed system.

6. Discussion

In this paper, we have shown how the customized IKEv1 program can cooperate with the novel smart card software. It is one application of the proposed simple device authentication framework. This application achieves peer to peer device-oriented authentication based on production-level identity described in section 2.1. A conventional token such as a PKCS#11 [12] compliant smart card focuses on human being authentication only. In contrast, our proposed framework and the application shown in this paper can handle device’s production-level identity such as the manufacturer’s ID and the product’s serial number. Furthermore, our original novel smart card software can provide owner-level attributes authorization APIs based on AC’s attributes and ACL rules. On the other hand, the current performance of the proposed middleware is not fast. However, the processing speed of the smart card hardware may improve in the near future, and thus the processing time of our developed software will also decrease for practical usage.

7. Conclusion

In this paper, we have given the prototype application of a framework not for use by human beings but for devices. We have employed smart card technology to append authentication functions to conventional devices such as home appliances. Our novel device-oriented authentication framework and its application shown in this paper achieves production-level identity authentication between two devices using a customized IKEv1 program. In our future work, we will develop the application to achieve the controlling of real devices such as home appliances under owner-level authorization rules using our smart card software. Our framework and its application shown in this paper will help to realize secure ubiquitous computing for home appliances.

Acknowledgement

The fundamental part of our smart card software was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Young Scientists (B), 17700082, 2005.

References


