2012 4th International Conference on Computer Engineering and Technology (ICCET 2012)
IPCSIT vol.40 (2012) © (2012) IACSIT Press, Singapore

A Kerberos-based UPnP Extension for Secure Home Networks

Huiya Zhu, Yuesheng Zhu+

Communication and Information Security Lab
Shenzhen Graduate School, Peking University, Shenzhen, China
huiyaz@sz.pku.edu.cn, zhuys@pkusz.edu.cn

Abstract. UPnP is a popular protocol in digital home network. However, no mature security mechanism is provided in current UPnP. In this paper, we propose a new UPnP extension based on Kerberos (KUPnP) to enhance the security of UPnP network. A Key Distribution Centre (KDC) is introduced to handle the mutual authentication between control points and devices. Also a security enhanced service "*SecureKUPnPService*" is provided for device template to perform secure handshake and build secure channel for network entities. Implementation results show that the control points and devices in UPnP home network can be authenticated with KUPnP and communications among all entities are secured.

Keywords: UPnP, network security, Kerberos, home network, authentication

1. Introduction

Universal Plug and Play (UPnP) [1] is an extensively accepted protocol for digital home network. It supports zero-configuration and auto-discovery by taking advantage of modern web technologies including HTTP, XML, SSDP, SOAP and GENA [2]. It brings much convenience to the home networks and enables flexible service managing and contents sharing between different devices. However, some existing problems have limited its development, and the most important issue is network security. UPnP does not directly specify any countermeasures for security threats in its origination; several potential vulnerabilities can be exploited. UPnP lacks authentication for devices, so that any control point can access all services and resource on any UPnP device. Also, messages delivered between UPnP devices and control points are in plaintext, which may result in exposure of home privacy.

Several approaches have been proposed to enhance the security of UPnP network. UPnP Forum defined the security architecture [3-5] by including two key modules: the "Device Security" for providing services to secure the UPnP Simple Object Access Protocol (SOAP) actions (i.e. authentication, privacy etc.), and the "Security Console" to offer administration of access control on security-aware UPnP devices. However, this architecture is a very complex structure and not a practical solution for UPnP vendors. In SUPnP, a secure communication method over UPnP networks developed in [6], two entities are imported: a key manager to maintain the relationships of devices and generate secrete keys, and a forwarder to relay the broadcast requests and convey the corresponding responses. SUPnP guarantees both the privacy and confidentiality of possible sensitive data transmitted in the network; however, it lacks authentication mechanism for devices. A secure UPnP network in [7] uses Transport Layer Security (TLS) to ensure privacy of most of the data transmitted in UPnP network, and a local Certificate Authority (CA) releases X.509 certificate to authenticate UPnP devices and control points. Also, "OpenHouse" [8], a TLS based distributed security architecture uses TLS to provide authentication and authorizations for UPnP home network, and devices can perform mutual authentication with each other by public keys in X.509 certificates. However, applying TLS [7-8] changes the underlying protocol of UPnP and gets complex for UPnP vendors to implement.

+ Corresponding author. Tel.: + (86)75526035352; fax: + (86)75526035352
E-mail address: zhuys@pkusz.edu.cn

In this paper, we propose KUPnP, a new UPnP extension based on Kerberos service [9] to construct a secure UPnP home network. Several essential functions of Kerberos are leveraged to achieve message flow's confidentiality and integrity in UPnP transaction, while tickets are applied for authentication of entities over UPnP network. A Key Distribution Centre (KDC) is introduced to act as a trusted centre performing authentication process. By importing a secure service "SecureKUPnPService", we provide a particular service template for creating service descriptions which encapsulate several key actions to execute authentication for control point and device, and then build a secure channel between the two end points.

The rest of this paper is organized as follows. The KUPnP's network architecture is described in Section 2. Design details are elaborated in section 3. Section 4 gives the evaluation of KUPnP. The paper is concluded in Section 5.

2. KUPnP Network

2.1. Notations for proposed KUPnP model

Table 1 presents a list of notations which are used in the authentication procedure of KUPnP networking.

Table.1 Notations for proposed KUPnP model

Notations	Description of the symbols
c	a kCP
K_c	kCP's secret key, hash of c's password
c_{addr}	IP address of c
tgs	ticket granting server, part of the KDC
K_{tgs}	tgs's secret key
$\{\}_{Ktgs}$	an encryption with the K_{tgs}
$K_{c,tgs}$	a session key shared between kCP and tgs
$timestamp$	indication of entity validity
TGT	Ticket-Granting-Ticket
d	a kDevice.
K_d	kDevice's secret key
$K_{c,d}$	session key shared between kCP and kDevice
$Ticket_{c,d}$	the ticket kCP uses to authenticate itself to kDevice

2.2. Entities in KUPnP Network

The KUPnP network contains the following entities: a Key Distribution Center (KDC), several kCPs and several kDevices.

kDevice: Secure UPnP device built on KUPnP layer. Each kDevice includes a compulsory service which is responsible to conduct the authentication process, several required security specific actions is defined into its service template (the XML service description template).

kCP: Secure UPnP control point built on KUPnP layer. KCP is the identified control point and has a clear identity to access the home network.

KDC: We introduce Key Distribution Center (KDC) as a trusted central manager to handle key management and authentication between devices. It consists of two logical parts: Authentication Server (AS) and Ticket-Granting Server (TGS). KDC maintains a database of secret keys of all kCPs and kDevices. For communication purposes, it generates a session key for device and control point to use to encrypt/decrypt their communications.

2.3. Working Mechanism

The KUPnP networking works at the following steps as shown in Fig.1:

Initialization: kCPs and kDevices are registered in KDC database first in the form of Kerberos principals. After registration, the kCP or kDevice obtains an address, and then can communicate with other devices on the network.

Discovery: When a kDevice is added to the network, it makes itself known to control points and advertises its services by sending NOTIFY messages. Similarly, the kCP multicasts M-SEARCH discovery message to search for interested devices and services.

Authentication: By sending an AS_REQ message to the KDC, kCP clarifies its identity before requesting any service on discovered device. With a correct password provided, a Ticket Granting Ticket (TGT) is returned by KDC that is used for later service invoking.

Handshake: In the secure UPnP handshake phase, a mandatory service "SecureKUPnPService" is requested first to issue a ticket to the kCP for desired kDevice, and then a session key is retrieved from the ticket to encrypt communication with them to ensure a secure session.

Interaction: After secure channel is established between the kCP and kDevice, all subsequent control and subscription actions between the two ends are encrypted, and thus guarantees secure interactions.

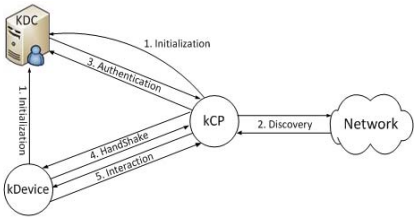

Fig. 1: Working mechanism of KUPnP

3. Design of KUPnP

3.1. Initialization and Discovery Module

In this context, we suppose kCP and kDevice are devices admitted by the host to participate in the home network. When a new kCP or kDevice joins the KUPnP network, it first registers in the KDC database in the format of a principal according to the configuration files from end user. The format for a Kerberos principal is primary/instance@REALM. Each entity in the network shares a secret key known only to itself and to the KDC. Knowledge of this key serves to prove an entity's identity. After registration, the kCP or kDevice obtains an address via Dynamic Host Configuration Protocol (DHCP) or automatic IP addressing (Auto-IP), and generates a port number for use.

In KUPnP, discovery works as the foundation for other phases. Discovery in kCP includes two parts: advertising M-SEARCH Simple Service Discovery Protocol (SSDP) messages or accepting NOTIFY messages from kDevice.

3.2. Get TGT in Authentication phase

After kCP gets a list of devices, it will invoke the interested services. Before that, it needs to clarify its identity to KDC by providing its password, and get a TGT, which will be used for kCP to request various services from specific kDevice in later interactions.

Here we describe the authentication details at the following steps:

Step 1: The kCP sends an *AS_REQ* message to the AS in KDC requesting authentication:

$$AS\_REQ: < c, tgs, n, timestamp >$$

Step2: The AS checks to see if kCP's principal is in its database. If it is, the AS sends back the following *AS_REP* message to the client:

$$AS\_REP: < \{TGT\}\ K_{tgs},\ \{K_{c,\ tgs}\}\ K_c >$$

$$TGT: < c, caddr, timestamp, K_{c,\ tgs} >$$

Step3: Once kCP receives message *AS_REP*, it attempts to decrypt $\{K_{c,\ tgs}\}\ K_c$ with the secret key K_c generated from its password. With a valid password and secret key the kCP decrypts $\{K_{c,tgs}\}K_c$ message and obtain the kCP/TGS Session Key $K_{c,tgs}$. This session key will be used for further communications with the TGS.

At this point, kCP already has enough information to authenticate itself to the TGS and then get access to request services.

3.3. SecureKUPnPService

In KUPnP, in order to carry out authentication with kCP, we define a mandatory service named "SecureKUPnPService" and add it to the <serviceList> in the description xml file of each kDevice. Three actions are defined in this service description xml, respectively named: "OpenSecureHandshake (Open_SH)", "BuildSecureChannel (Build_SC)", and "CloseSecureHandshake (Close_SH)".

In KUPnP model, “SecureKUPnPService” handles the Handshake Phase and is a key service for building the secure channel between kCP and kDevice. Action “Open_SH” is first invoked by kCP to open a secure handshake with the kDevice, resulting with a *HandshakeID* allocated by calculating the MD5 of the kCP name. HandshakeID stands for a kCP in handshake phase and facilitate kCP to be located easily. “Build_SC” action plays important role in this phase. It accomplishes two major procedures: the first is to enable kCP to get an authorized CP/Device ticket from TGS, and the second is to accomplish mutual authentication with kDevice using the ticket. Then a secure channel is built between kCP and kDevice, which ensures security for later data transmission. When the Build_SC action is invoked, following steps take place in sequence:

Step1: kCP sends a *TGS_REQ* message to the TGS to request services from the wanted device:

$$TGS\_REQ: < \{TGT\}_{Ktgs}, d, Authenticator >$$

$$Authenticator: < \{c, timestamp\}_{Kc,tgs} >$$

Step2: The TGS retrieves TGT out of message *TGS_REQ* and decrypts it using the TGS’s secret key K_{tgs}. This gives it the "kCP/TGS session key" $K_{c,tgs}$. Using $K_{c, tgs}$, the TGS decrypts the Authenticator and sends a *TGS_REP* message to the client:

$$TGS\_REP: < \{Ticket_{c,d}\}_{Kd}, d, \{K_{c, d}\}_{Kc,tgs} >$$

$$Ticket_{c, d}: < c, c_{addr}, time', K_{c,d} >$$

Step3: Now the kCP has a ticket to authenticate to the desired device. It connects to the kDevice and sends the following *AP_REQ* messages with a new authenticator in it:

$$AP\_REQ: < \{Ticket_{c,d}\}_{Kd}, d, Authenticator' >$$

$$Authenticator': < \{c, timestamp\}_{Kc, d} >$$

Step4: The kDevice decrypts the ticket using its own secret key K_d to retrieve the CP/Device Session Key $K_{c,d}$. A malicious device without the right password will fail to decrypt it. Using the $K_{c,d}$, kDevice decrypts the new *Authenticator'* and sends the message *AP_REP* to the kCP to confirm its true identity and willingness to serve the kCP:

$$AP\_REP: < \{timestamp+1\}_{Kc, d} >$$

Step5: The kCP decrypts the confirmation message by using the CP/Device Session Key $K_{c,d}$ and checks whether the timestamp is correctly updated. If so, then the control point can trust the device and can complete mutual authentication then start issuing UPnP control requests to the device.

“Close_SH” action is finally invoked to close the secure handshake, then kCP and kDevice enter secure KUPnP interaction phase.

3.4. Secure KUPnP Interaction

Now that a secure pipe has been established between the kCP and kDevice, mutual authentication has also been conducted, so the devices and control points can carry out further interactions (including control, event, and presentation) with each other.

To control a device or to invoke actions on device’s services, a control point sends control message to the control URL contained within the device description XML for the KDevice’s service. Control messages are also expressed in XML using SOAP. In response to the control message, the device or service sends results (action response, query response) to the control point. The service publishes updates by sending event messages when its variables change, and a control point can subscribe to receive this information.

4. Evaluation

To verify the proposed KUPnP model, a prototype of the network is implemented. We develop an UPnP control point with name “Alice” and UPnP device named “Bob” respectively over Linux platform by leveraging Intel UPnP SDKs and source code of krb5-1.9 released by MIT Kerberos team. The experiment is carried out in a testbed network which simulates a typical home network, as shown in Fig.2. Alice is new to the home network, by providing the right password and having authenticated with KDC, a TGT is generated. Then Alice makes use of this TGT to obtain a ticket for specific service and actions from kDevice Bob. Generated TGT and ticket can be seen in Fig3. In this case, a TGT’s valid time is set to 24 hours, which means during this period, this TGT can be used by Alice to ask for services at any time, not need to give

password again. The valid time of TGT can be adjusted according to home user's configuration. With a valid ticket, Alice is able to invoke actions and conduct further secure communications with Bob.

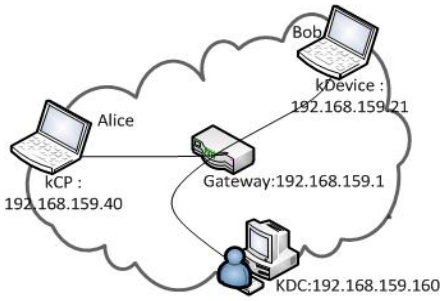

Fig. 2: Testbed for KUPnP network

```
Ticket cache: FILE:/tmp/krb5cc_1000
Default principal: Alice@SZ.PKU.EDU.CN

Valid starting     Expires            Service principal
02/06/12 15:44:24  02/07/12 15:43:29  krbtgt/SZ.PKU.EDU.CN@SZ.PKU.EDU.CN
02/06/12 15:44:36  02/07/12 15:43:29  Bob@SZ.PKU.EDU.CN
```

Fig. 3: Output tickets for kCP "Alice"

Take a sample action "GetDoc" in service "MusicServer" as example, Alice invokes this action to get a document from Bob. Fig.4 shows the body of SOAP messages for response of action "GetDoc", sniffered by network protocol analyzer Wireshark. The string between <HandshakeID> field is the MD5 value of kCP's name "Alice", uniquely identifing a control point in the interactions. Text between field <OutDocBody> are

```
<u:GetDocResponse xmlns:u="urn:schemas-upnp-org:service:MusicServer:1">
    <HandshakeID>9265043d3ed60c298bf724d7bfede7e6</HandshakeID>
     <OutDocBody>dYGJMIGGoAMCAQWhAwIBFaN6MHigAwIBEqJxBG/pZN
```

the encrypted messages transmitted between them.

Fig. 4: Encrypted document body of "GetDoc" action in KUPnP

5. Conclusions

In this paper, we analyze the key security threats in current UPnP and propose a new UPnP extension named KUPnP to obtain a secure and efficient UPnP network. We take advantage of Kerberos service and set up security-enhanced mechanism to guarantee the network message's privacy and mutual authentication between devices. By implementation, we verify that our approach is effective in mutual authentication between UPnP devices and control points, messages are well encrypted. What's more, we also show that KUPnP is realized as a single sign on system which reduces frequent human intervention after secure session is set up and therefore keeps UPnP's zero-configuration to the most.

6. References

[1] UPnP Forum, "UPnP® Certified Technology—Your Simple Solution for Home, Office and Small Business interoperability", Sep. 2010

[2] B. A. Miller, T. Nixon, C. Tai, M. D. Wood, "Home networking with universal plug and play," IEEE Communications Magazine, vol. 39, no. 12, pp. 104–109, 2001

[3] C. Ellison, "UPnP security ceremonies design document for UPnP device architecture 1.0", UPnP Forum, Oct.2003

[4] C. Ellison, "SecurityConsole:1 service template for UPnP™ device architecture 1.0", UPnP Forum, Nov.2003

[5] C. Ellison, "DeviceSecurity:1 service template for UPnP™ device architecture 1.0", UPnP Forum, Nov.2003

[6] J. Lee, C. Huang, L. Lee, and C. Lei, "Design and implementation of secure communication channels over UPnP networks", in Proc. MUE, pp.307-312, 2007

[7] V. Pehkonen, J. Koivisto, "Secure universal plug and play network", in IEEE International Conference on Information Assurance and Security (IAS, pp.11-14), 2010

[8] J. Suomalainen, S. Moloney, J. Koivisto, and K. Keinänen, "OpenHouse: a secure platform for distributed home services", in Proc. PST, pp.15-23, 2008

[9] B. Clifford Neuman and Theodore Ts'o, "Kerberos: an authentication service for computer networks", IEEE Communications Magazine, Volume 32, Issue 9, pp.33-38, Sep.1994.