Ubiquitous Authorization Scheme Based on Device Profile

Kevin Tham, Mark Looi, and Ernest Foo

Information Security Research Centre,
Queensland University of Technology,
GPO Box 2434, Brisbane, QLD 4001 Australia
{wk.tham, m.looi, e.foo}@qut.edu.au
http://www.isrc.qut.edu.au

Abstract. The range of devices that are capable of connecting to data networks has been on a rise in recent times. From the perspective of an administrator, controlling access to data networks, via these devices, usually includes the creation of separate login credentials. This leads to an administrative nightmare, from both the user and administrator's point of view. This paper proposes a novel approach to this problem and offers a single-sign-on system, where the user's authorisation is based on the login credentials of the user, and the profile of the device the user is using. An instance of this design is presented with SESAME, to demonstrate the usefulness of the design, and also practicality for implementation.

1 Introduction

Up until recently, devices capable of connecting to data networks have been similar in nature; usually computers with a capable processor and large storage. Advancement in device connectivity has introduced a diverse range of mobile devices that are able to access network resources, and with this comes the risk of weakening the overall structure of security in the network.

One scenario would be the job of a network administrator. The administrator may log in from his workstation in his office, running daily tasks to up-keep the entire network. However, an administrator does need to go on-site to solve certain network problems, and might carry a PDA, connected to the network via the wireless LAN. From the PDA, the administrator will have a set of tools that will aid him in his work. When the same administrator heads off-site, he might have a mobile phone that connects him to a system, that tells him the status of networking services, through the GPRS network. Giving him constant updates on the status of each network service. To solve this scenario, multiple accounts have to be made for every device the administrator uses, so that the lost of one of the device, will lessen the security impact to the network security.

This paper will propose a generic design for a ubiquitous authorisation scheme based on a device profile. This scheme was designed, so that current secure network designs are not impacted in a large way. This will lead on to an instance of this design being presented with SESAME.

X. Zhou et al. (Eds.): EUC Workshops 2006, LNCS 4097, pp. 312–321, 2006.

2 Related Work

Deriving a user's sub-set of access, based on the device they are logging in from, does involve several considerations, but is not a difficult concept. The more challenging area, will be determining the device, with which the user is logging in from. Device identification can be determined either automatically, or through a negotiation phase.

A method that involves a negotiation phase would be the Extensible Authentication Protocol (EAP) (RFC2248 [BV98]), which is part of the Point-to-Point Protocol (PPP) (RFC1661 [Sim94]). When a PPP session is set-up, the client negotiates the authentication method. The negotiation for authentication, is based on the availability of the method, rather than the ability of the client in handling certain authentication methods. While simplistic in nature, it is effective, though allowing the client to determine the device's capabilities is not ideal. The Transport-Layer Security Protocol (TLS) [DA99] takes a similar approach to EAP. During the TLS Handshake procedure, both the client and server will agree upon a protocol version, select cryptographic algorithms and optionally, authenticate each other. Much like EAP, TLS had been developed to accommodate a wide variety of authentication mechanisms, and in the specific case of TLS, cryptographic algorithms. This sort of authentication will depend on what the authenticating device is capable of, therefore a kind of profiling for the devices.

An example of an automatic process for profiling a device is discuss in the implementation used in the Java 2 platform Micro Edition (J2ME) design. The Connected, Limited Device Configuration (CLDC) [Sun00] $HotSpot^{tm}$ [Sun02] implementation boasts a subset of functions from the Connected Device Configuration (CDC) [Sun01] profile. The CDC uses the Java Virtual Machine (JVM) interpreter, whereas the CLDC utilises the K Virtual Machine (KVM). Because of the nature of devices found in CLDC, KVM was designed to have a small footprint. This meant that it offers a subset of features to that of the JVM. This does not relate directly to a security architecture, but profiling devices based on what it can handle is introduced as a possible profiling technique.

3 Security Requirements

The security requirements for a ubiquitous authorisation scheme should encompass the following attributes; Single set of login credentials, device profiling, multi-level categorisation of access control, and least impact on current secure network architectures.

3.1 Single Set of Login Credentials

A single set of log-in credentials offer very similar attributes to a single sign-on system. The aim of a single sign-on system, like SESAME [PP95], is to offer access to multiple services within a single network without the need to re-authenticate. This is transposed to this requirement, whereby users are allowed to gain access to a data network using a single log-in credential, as opposed to separate log-in credentials for every device they want to gain access to the network from.

3.2 Profiling Techniques

Profiling devices involves the step of categorising the different attributes and abilities that a device can handle. This will pertain to the level of "trust" a device has on the network. It is through the use of profiles, that allow for a clearer view on how much "trust" a device might be imparted with. It also provides for the identification of devices that may connect to a network. The end result of profiling will include a set of devices that fall into different categories. This allows the convenience of handling a group of devices as one, thereby reducing complexity. This method also allows for future additions, and offers flexibility. Profiling devices is further discussed in Section 4.2, where methods of automatic and manual device identification are explored. This is first preceded with a discussion on how devices can be profiled.

3.3 Multi-level Categorisation of Access Control

Categorisation of the access control list pertaining to a user is necessary to maintain a standard list of access control across the organisation. This allows for access control entries to be placed in different categories. For example, access to administrative tools like password changing and user profile update, could fall under one category, and web-access with proxy rights could fall under another category. This allows for ease of administration, however, the ability to fine tune the access control of each system should still be an option.

3.4 Access Granting

When a device has been profiled, it is up to the authorisation server to decide on how much access the device is to be granted. There are two approaches to this problem; allowing full access, and then limiting it, or giving only the right amount of access.

Both methods are considered to be equally secure, however, emphasis is placed on the amount of processing required. For the first method, the network will have to depend on another system to reduce the access control list down to size, whereas for the latter method, dependence is placed on the authorisation server to process the correct amount of access to grant and not give anymore than it should. This approach works well in an environment that offers Access Control Lists (ACLs) to user during authorisation. However, this does not apply to systems that depend on role-names or Role-Based Access Control (RBAC) [FK92] systems [MTHZ92, Gan95, KN93].

Roles call for a different approach to this requirement. The ACLs in these systems are not stored in a central repository for retrieval, but rather, every network service contain rules on which roles are authorised for connection. In this case, either the system has to be able to discern the difference between devices automatically, or an intermediary system could change a user's profile, to suit the device the use is logging in from.

3.5 Least Impact on Current Secure Network Architectures

The aim for the proposed system, is to have the least impact on current secure network architectures. This is to ensure a wider acceptance of the technology, whilst still

preserving the current rollout of the network architecture that an administrator has. Consideration has to be placed into the integration of the system. One common way is to have an intermediary proxy in place.

4 Proposed Generic Design

The proposed design consists of the usual components which are found on a security architecture. These include the following: The *User*, *Log-in Device*, *Authentication Server*, *Authorisation Server* and the *Network*.

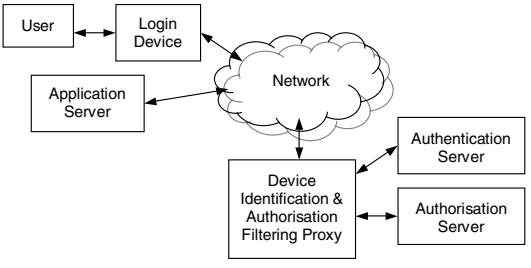

Fig. 1. Proposed Network Design

4.1 Overall Operations of Proposed Generic Implementation

The *User* is any person with a legitimate account on the network. They possess the *Log-in Device* that will allow them to access the network resources. The user can only produce their credentials for access to the network, through the aid of the Log-in Devices. These devices refer to a large number of networked devices, ranging from simple Personal Digital Assistants (PDAs) to full-fledge workstations, making this portion of the network heterogeneous. Although the devices are assumed not to be trusted by the network, an assumption has to be made that the module that connects to the network security framework, is trusted. This module resides in the log-in device and will have the task of proving its identity. There are several of doing this, and are outlined in Section 4.2.

The log-in device will now act on behalf of the user, to communicate to the security server via the *Network*.

The first point of contact for the log-in device will be the *Device Identification and Authorisation Filtering Proxy*. This proxy is put in place to handle the communication between the user and the security server. This will impact the least on existing security architectures. At this stage, the proxy will handle the authentication of the device. This allows the proxy to tag the message from the log-in device to the *Authentication Server*. Placing the proxy at this point allows for an additional feature to security. With the device authentication in the message, the authentication server can now decide if a user has access on the network, and also if the user has rights to authenticate through certain devices. When the authentication server is done, it will contact the *Authorisation Server* next. This is done via the proxy again, but this time, the proxy just forwards the message, and will not manipulate it.

The authorisation server should function as normal, and the next message should be sent from the authorisation server to the log-in device. This message will contain the user's ACL. But before it gets received by the log-in device, it will have to pass through the proxy again. This time, the proxy will act as a filter and decide on the level of access a user should have, based on the device they have logged in from. This sub-set of the ACL is then returned from the proxy, to the user.

4.2 Device Profiling

Developing a level of trust based on the capabilities of a device is one of the ways in identifying the device. Many devices have differing processing and storage capabilities. This ranges from simple 16MHz processors found commonly in PalmPilot PDAs to 3GHz processors on desktop computers, and storage that ranges from a few hundred kilobytes to gigabytes or even terabytes of storage. The aim of this is to be able to profile a device based on its capabilities to handle cryptographic calculations. Handling the identification of devices by profiling the cryptographic abilities has a direct relation to the ability of the device to be secure. However certain devices do offer better implied security, without the need to have a powerful processor or large storage. Instead, these devices are tamper-resistant in nature, like a smart-card. This means that the profiling will start from the top, between tamper-resistant devices and non-tamper resistant devices. This list will be further broken down into different capabilities of the devices.

The profiling of a device can be handled in several ways. However, two common techniques included are an automatic device profiling and a manually profiling technique. Both techniques offer merits and drawbacks at the same time. The biggest consideration has to be the implication to communicating protocols and also complexity of design.

The automatic device profiling technique offers the most flexibility, in terms of identification of currently available devices and future offerings. A novel approach is to measure a cryptographic calculation in terms of the device's response time. This measurement is then compared against a set of known response times, thereby deriving the device's identification. This concept is simple, but a complex set of protocols is needed to handle it. Other unknown factors like processor bottleneck, and even transmission medium congestion have to be taken into consideration. This is an immense task for deriving a non-guaranteed value, measured in milliseconds.

The other method for profiling happens manually. All devices wanting to be used to access network resources will have to be identified. All results are placed in a central repository, with proper identification tags placed in a module of the device. This approach allows for much better use of the network, and also requires a less complex protocol. However, profiling every single device on a network will prove to be a tedious job.

With both profiling techniques in mind, choosing the manual technique seems to be a more efficient way of designing the protocol. Allowing for large overheads in communication of a network is best avoided this way. Although tedious to "register" every device, it does not have too many factors to consider when deciding on its access level. Manual profiling can be based on the following attributes.

Table 1. General Device Profiles

Hardware	*Description*
Slow Processor, Small Memory	These devices are generally considered to be the least secure. This usually implies that the device will not be able to handle a strong cryptographic scheme.
Slow Processor, Large Memory	Devices will be able to handle cryptographic schemes with larger key sizes, and are considered to be somewhat more secure, compared to the last category. Handling cryptographic schemes with large key size will take an impact in the processing time.
Fast Processor, Small Memory	With a faster processor, these devices are considered to be more secure than the last two categories. However, the devices in this category will be limited to schemes with smaller key sizes.
Fast Processor, Large Memory	These devices are generally the most secure amongst the other categories. There will not be any limitation to the key size and processing times.

5 Instance of Proposed Protocol with SESAME

The proposed design aims to impact as little as possible to existing security architectures, but yet extends it with a mutli-level authorisation scheme. This is to be illustrated with the presentation of the proposed design in SESAME. SESAME has been chosen, because of its well-rounded, and complete approach to an authentication and authorisation architecture. This allows for better understanding of the concept of the proposed system.

5.1 Proposed SESAME Extension

Based on the proposed design in Section 4, implementation is to take the form of a proxy-type service. The main task of this proxy is to intercept message exchanges during authentication and authorisation, so that appropriate device identification and authorisation filtering can be done. However, based on current SESAME standards, it is not necessary to include the proxy-type service to messages between the client and the *Security Server*. The biggest change in SESAME will be the inclusion of new message fields in the communication messages. This allows for an easier approach of writing plugin modules.

Figure 2 shows the addition of four plugin modules to the various components to the original SESAME architecture. These are the *Device ID Module* found in the Client, the *Device Authentication Module* found in the Authentication Server (AS), the *Device Authorisation Module* found in the Privilege Attribute Server (PAS), and the *Device ACL Module* found on the individual Secure Association Context Manager (SACM) of the target servers.

The following is an outline of the changes made to SESAME, extended to make use of the proposed Multi-Level Authorisation design.

1. During the authentication phase, the APA Client will generate a KRB_AS_REQ, as specified by RFC1510. This does not differ from the Kerberos message structure, as designed in SESAME. The Device ID Module found in the

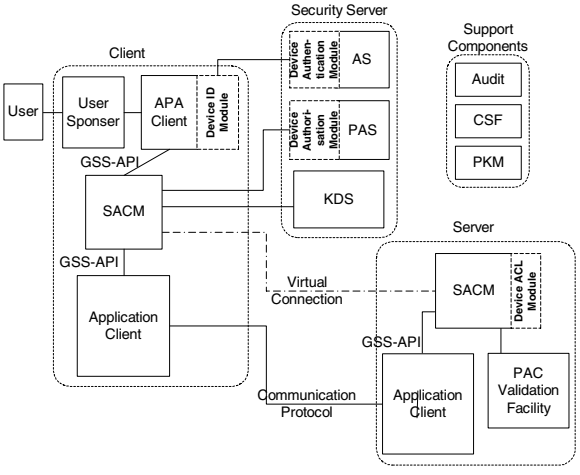

Fig. 2. Proposed Changes to SESAME

APA Client will then modify this structure, by including a Device ID field, identifying the device the APA Client resides in.

2. On receiving the first message, the *Authentication Server* (AS) will first authenticate the user, as specified in the original SESAME design. Once user authentication is obtained, the Device Authentication Module will then authenticate the device. This module will act as an additional level of authentication, whereby it will decide if a user has authentication rights on certain devices. If the user is not allowed to log in from the device, an error message is generated by the AS, informing the user of a "failed to log in" message. This module will place the Device ID into an additional field in the PAS Ticket, returning from the Authentication Server to the APA Client. The logic of this module is as follows:
3. When recieved by the APA Client, the PAS Ticket found in the KRB_AS_REP is cached, after which the APA Client will request for a Privilege Attribute Certificate (PAC) from the *Privilege Attribute Server (PAS)*. This is done through the initiator's *Secure Association Context Manager (SACM)*. The SACM sends the PAS Ticket to the PAS for authorisation. At this point, the PAS Ticket still contains the Device ID field.
4. Authorisation is handled as specified in the SESAME standard. The Device Authorisation Module found in the PAS only ensures that the Device ID field is transposed onto the corresponding PAC, created by the PAS. This will allow for the identity of the device to be made known to the target server. The PAS then builds the PAC (with the inclusion of the Device ID field) and signs it. The PAS is then returned to the initiator's SACM. It is important to point out that since SESAME handles authorisation based on role names, an exception is made to the functionality of the device authorisation module. In the instance where an authorisation message contains an ACL, the device authorisation module will reduce the set of access, based on the device's profile. However, role names cannot be reduced. Therefore the final decision on user access rights is handled by the targetted servers.

5. When the user wishes to access a server in the network, it will invoke the SACM to contact the targetted server's SACM. The initiator's SACM will send the PAC to the targetted server's SACM, which will in turn validate the PAC in-accordance to the SESAME standard. However, before further communication is allowed between the applications, the Device ACL Module will do the final check. In SESAME, the ACLs are stored on the individual SACM of target servers. The SACMs will handle all the communication, and will base its decision on the user's role, found in the PAC. This module extends this decision making with an additional deciding factor; access to the targetted server via a specific device. After the SACM has gone through all the deciding factors, this module will then have the last say as to whether a user has access to it. It contains an additional ACL for devices, mapped against a list of users on the system. Access rights are handled as follows:

Table 2. System Access Rights Based On Device Rights

	Access Rights for Devices	*No Access Rights for Devices*
Access Rights for User	**YES**	NO
No Access Rights for User	NO	NO

Table 2 shows that the only way a user can have access to a target system, is if the device is allowed access to it.

5.2 Proposed SESAME Message Changes

Not much will be changed, in terms of message structure. This is to minimise impact on currently running systems. The addition of fields to tickets and tokens are the most drastic changes.

The following description of changes to messages are in reference to the ECMA-219 [ECM96] specification of a PAC used by SESAME.

Message 1 consists of the KRB_AS_REQ message, and a strong authenticator. This message is used to obtain authentication credentials for the user. The strong authenticator is present because a public-key extension to the authentication phase in SESAME is assumed. Otherwise, the strong authenticator will be absent. The KRB_AS_REQ message structure follows the RFC1510 [KN93] standard. Modification to the message is the inclusion of a Device ID field, added by the Device ID Module as outlined in Section 5.1. This Module is assumed to be trusted in placing a correct device identification tag. The addition of a `device-id[12]` field in the `KRB-REQ-BODY` allows for a string, containing the type of device the user is logging in from. The possible values might be Workstation, PDA, MobilePhone, NoteBook.

Message 2 is the return message sent from the authentication server to the client. The structure of this message is exactly the same as that of the KRB_AS_REP message found in RFC1510 [KN93]. This includes the PAS Tic-ket, as specified as a Ticket-Granting Ticket (TGT) in RFC1510, and some Kerberos Control Data. The PAS Ticket is used to obtain authorisation data, in the form of a PAC, from the PAS. An addition of

a `device-id[11]` in the `EncTicketPart`. The `device-id[11]` field contains the device identification tag that was obtained from the first message.

Message 3 contains the PAS Ticket obtained from the last message. This Ticket has no modification on it, so the `device-id[11]` in the `EncTicketPart` is still preserved. This message is sent to obtain a PAC from the PAS, based on the *user's role name*.

Message 4 is sent from the PAS and back to the Client. This message contains the user's PAC, signed by the PAS. The PAC contains the `device-id[9]` in the `PAC SpecificContents` which has been obtained from `deviceid[11]` in the `EncTicketPart` from Message 3.

Message 5 may be sent out immediately on receipt of message 4 or much later, but within the life-time of the PAC. The PAC is sent to any targetted server, to obtain access rights into the system. This message contains all the information from message 4, including the `device-id[9]` in the `PACSpecificContents`. This allows for the Device ACL Module (as outlined in Section 5.1), to make its decision on the access rights of a contacting user.

6 Conclusion

In this paper, we have discussed the merits of having a ubiquitous authorisation scheme based on a device's profile. Advantages of this design are more evident with the proliferation of mobile devices that have connectivity to data networks. Instead of solving this problem by adding multiple log-ins, the proposed authorisation scheme allows a user to log-in to the network from any device, while using the same set of credentials. This is a convenient method for users, as well as administrators. On the one hand, users do not need to remember different passwords for different systems. On the other, administrators have a scheme that allows a finer granularity in control over device connections, minus the tedious work of creating multiple log-ins for everyone.

One of the main disadvantages pointed out in this paper, is that devices still need to be manually profiled. The idea of an automatic profiling system is a novel idea to solve this problem, but runs into the risk of false-positives and false-negatives. This is especially so, since measurement is done in milliseconds, and too many varying factors could impair the calculation. One other lacking in feature is the protection of the device identification tag in each message. A possible implementation, in this case, could be a signed tag of the device name, by the administrator. This in turn can be verified by the Public-Key Infrastructure (PKI) of a network.

Future work in this research, could encompass the idea of another profiling technique based on the traversal pathway of a communication message. This allows for the network to be able to determine if the communication has crossed a segment of a network which does not have implicit trust with the secure portion of the network. Work done in IPv4 [Pos81] includes the idea of a source route, but does not offer an audit of where the packet actually transmitted through. If a profile of transmission path is possible, then this research will include an extra factor, whilst deciding on the authentication of a user.

References

[BV98] L. Blunk and J. Vollbrecht. *RFC2284: PPP Extensible Authentication Protocol (EAP)*, March 1998.

[DA99] T. Dierks and C. Allen. *RFC2246: The TLS Protocol – Version 1.0*, January 1999.

[ECM96] ECMA International, 114 Rue du Rhône, CH-1204 Geneva, Switzerland. *Authentication and Privilege Attribute Security Application with related key distribution functions*, 2nd edition, March 1996.

[FK92] D. Ferraiolo and R. Kuhn. Role-Based Access Control. In *15th National Computer Security Conference*, 1992.

[Gan95] R. Ganesan. Yaksha: Augmenting Kerberos with public key cryptography. In *Internet Society Symposium on Network and Distributed System Security*, pages 132–143, February 1995.

[KN93] John T. Kohl and B. Clifford Neuman. *RFC1510: The Kerberos Network Authentication Service (V5)*. Digital Equipment Corporation, USC/Information Sciences Institute, September 1993.

[MTHZ92] Refik Molva, Gene Tsudik, Els Van Herreweghen, and Stefano Zatti. KryptoKnight Authentication and Key Distribution System. In *European Symposium on Research in Computer Security (ESORICS)*, pages 155–174, 1992.

[Pos81] Jon Postel. *RFC791: Internet Protocol*. Information Sciences Institute, University of Southern California, September 1981.

[PP95] Tom Parker and Denis Pinkas. *SESAME V4 – Overview*, December 1995.

[Sim94] W. Simpson. *RFC1661: The Point-to-Point Protocol (PPP)*, July 1994.

[Sun00] Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA. *Connected, Limited Device Configuration: Specification Version 1.0a, Java 2 Platform Micro Edition*, May 2000.

[Sun01] Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA. *Connected Device Configuration (CDC) and the Foundation Profile*, 2001.

[Sun02] Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA. *The CLDC $HotSpot^{tm}$ Implementation Virtual Machine, Java 2 Platform Micro Edition*, 2002.