Security and Privacy in a Middleware for Large Scale Mobile and Pervasive Augmented Reality

Pedro Ferreira, João Orvalho and Fernando Boavida
Centro de Informática e Sistemas, Department of Informatics Engineering, University of Coimbra
Polo II, Pinhal de Marrocos, 3030-290, Coimbra, Portugal
E-mail: {pmferr, orvalho, boavida}@dei.uc.pt

Abstract: **Ubiquitous or pervasive computing is a new kind of computing, where specialized elements of hardware and software will have such high level of deployment that their use will be fully integrated with the environment. Augmented reality extends reality with virtual elements but tries to place the computer in a relatively unobtrusive, assistive role. In this paper we propose, test and analyse a security and privacy architecture for a previously proposed middleware architecture for mobile and pervasive large scale augmented reality games, which is the main contribution of this paper. The results show that the security features proposed in the scope of this work do not affect the overall performance of the system.**

1. INTRODUCTION

A significant requirement of pervasive applications is fast service development and deployment [1], which implies the introduction of various service and application frameworks and platforms. For this, middleware is a common solution. The benefits of middleware utilization are the improved programming model, and the hiding of many implementation details, which make middleware based application development much faster. It is now becoming quite clear that entertainment, and more specifically mobile gaming, will be one of the killer applications of future wireless networks [2].
Augmented reality extends reality with virtual elements while keeping the computer in an assistive, unobtrusive role [3]. It is possible to create games that place the user in the physical world through geographically aware applications. Most of the latest mobile phones are equipped with cameras and some of the latest ones are coming with some form of 3D rendering technology [4] [5]. Bluetooth technology and increasing miniaturization will lead, in the near future, to low-cost, specialized pervasive equipment for augmented reality. In [6] we described the main objectives of our research concerning systems that satisfy the requirements of network middleware for large scale mobile and pervasive augmented reality games. In [7] we described a middleware system that is being developed for large scale mobile and pervasive augmented reality games that satisfies these objectives. The system targeted by the middleware is composed of 3 levels: the back-office central level, the large scale network level, and the personal area network level.
The full architecture of this system is described in more detail in [8], so here we just going to worry ourselves with describing the security and privacy issues and the solutions we added.
This paper focuses on security and privacy issues of the middleware proposed.
The main contribution of the paper is a security and privacy architecture for a middlware for mobile and pervasive large-scale augmented reality games.
The main objective of this paper is to show that while security and privacy is achieved with this architecture it does not adversely impact the performance of the whole system, and running augmented reality applications on top of it is still possible.
The paper is divided in Introduction (this section), Security and Privacy, Testing, and Conclusions, aside from abstract, keywords, acknowledgments and references.

3. SECURITY AND PRIVACY

Security and privacy are issues of significant importance on our middleware architecture for mobile and pervasive large scale augmented reality games.
Many times the information that is passed on the system should be kept secret for any other purpose other than gaming, and for any one else than the appropriate other players.
A striking example of this information is the location and orientation context information of the central device on the personal area network.
Of course this information needs to circulate on the system, but needs to be kept secure from outside attack.
So, we need encryption on the system. We also need authorization and authentication, to know which players and which devices can be connected.
Its all that architecture we be discussing in this section, for all the levels of the system.
We have built an architechture of security on our system that goes a step beyond and effectively extends the 3GPP security architecture (it is meant to work along side with it) [9][10] .

3.1 Personal Area Network Level Security

In the personal area network level of the system, we have a network of sensors and actuators that is connected to the central device through Bluetooth[11][12] and a central device that is connected to a large scale distributed level server through the use of TLS/SSL[13] over TCP.
So, the security we apply here is the following, we demand that all the Bluetooth connections be authenticated and encrypted. We apply security certificates do ensure

authentication and authorization in TLS/SSL over TCP and the encryption itself (witch is RSA).
This is all possible without using nothing more than Java capacities in J2ME and J2SE, including the capacity to install security certificates.
In doing so, we secure communications in the personal area network and in the communications between the personal area network and the large scale distributed server level.

3.2 Large Scale Distributed Level Security

On the large scale distributed server level, we communicate between servers using sixrm reliable multicast[14] using ipv6 [15], trough the use of our updated ARMSV6 corba event system using multicast, that evolved from previous work in ARMS – Augmented reliable corba Multicast System [16][17].
To be secure, we now symmetrically encrypt all communications that go trough ARMSV6 (and sixrm), in RCA5.
Communications are symmetrically encrypted (and not asymmetrically) because sixrm is an any-to-any multicast protocol and so there is no direct correspondence between the sender and the receiver.
The key of encryption is distributed by the central server to the distributed servers in the authenthication process.

3.3 Central level security

The central level is responsible for distributed server authenthication and authorization.
Every distributed server must know a login and password to the distributed server so that it can access this server trough TLS/SLL over TCP so it can receive the encryption key to use on symmetric encryption.
The key is passed encrypted over the secure channel. Certificates for this are pre-installed on the Java Virtual Machines the servers run on (J2SE).

4. TESTING AND ANALYSIS

To test our security architecture, we are going to test its usability for our target middleware objectives, which are mobile and pervasive large scale augmented reality games.
These objectives put scalability constrains on security architecture and also QoS constraints such as delay and jitter.
We can test the security architecture for delay and jitter targets, and we can analyse it to derive conclusions about its scalability.
We can divide these tests into the various parts of the system, namely in the personal area network, on the link between the personal area network on the large scale network level, and in the large scale network level itself.
This is because the central level works essentially as a key distribution level and so does not get itself involved in any realtime communications.

4.1 Personal Area Network Level Security Testing

The personal area network part of the system, both the STF PAN API and the STF SENSACT API, was subject to extensive functional and performance tests, with various kinds of simulated sensors and actuators and a simulated reading and actuating application using Java Wireless Toolkit 2.5 Beta from Sun Microsystems running in a series of emulators in a Pentium 4 3.6 GHz System with 1 Gb Memory. These tests give the same results as the tests run in [18] as nothing has changed, we still use Bluetooth encryption and authenthication. So, we do not present graphics here due to lack of space. So, we did not add any delay or jitter to the previous architecture.

4.2 Testing the communications between the PAN and the Large Scale Distributed Level Server

Between the personal area network central device, that runs Java 2 Microedition Mobile Information Device Profile 2.0 over the Connected Limited Device Configuration 1.1 (MIDP 2.0 and CLDC 1.1), and the distributed server where it happens to be connected there is a TLS connection over TCP where STF messages are exchanged according to our protocol.

To be able to test for delay and jitter on this connection, which is handled by specialized classes of STFPAN (the library of classes for the central device of the personal area network) and STFServer (the library of classes for the distributed servers and central server), we implemented timestamping of messages with the current time when sendingthe message, and automatically calculating the delay based on that timestamp and the current time on the receiving machine when receiving the messages. For the test to be meaningful, both machines must be synchronized through NTP, preferably to a common timeserver.

We also implemented logging to a file both on STFPAN and STFServer of the received delay values so that we can calculate the delay and jitter values for the various UE (mobile terminals or central devices of the personal area network), and elaborate graphics.

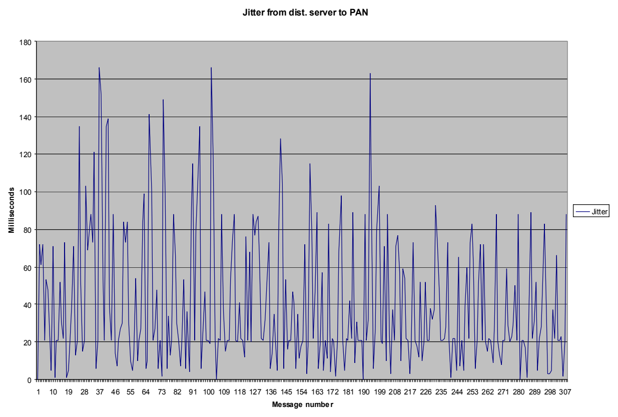

Figure 1 - Delay received at PAN from distributed server

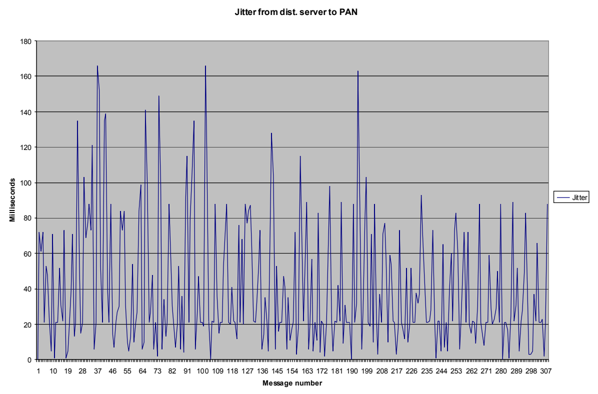

Figure 2 - Jitter received at PAN from distributed server

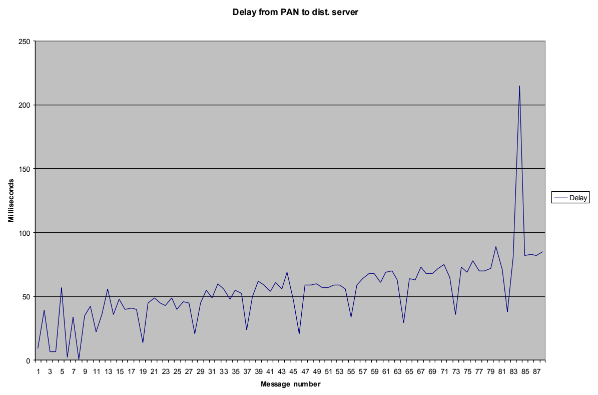

Figure 3 - Delay received at the distributed server from the PAN

We then made the tests with only one UE connected to one distributed server working in connection to one central server. In this situation, even if we do not have a working application to test messaging, there are messages exchanged between the distributed server and the central device of the personal area network, between the central device of the personal area network and the central server and between the distributed server and the central server.

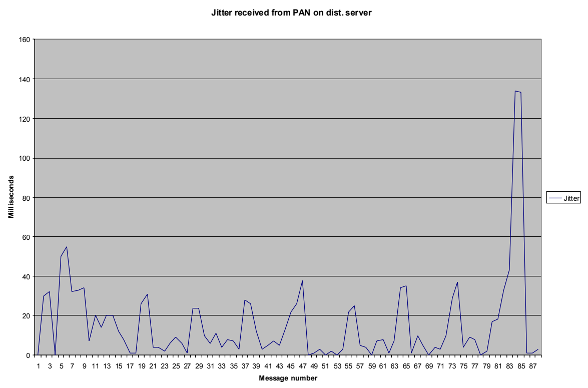

Figure 4 - Jitter received on the distributed server from the PAN

The messages we are interested in are the messages between the central device of the personal area network and the distributed server. At this stage, those messages are only messages of virtual time synchronization of STF's internal virtual time synchronization mechanism, which keeps a virtual clock synchronized between the central device of the personal area network and the distributed server. Is the delay and jitter of those messages that is evaluated in the graphics shown on this section.

Figure 1 shows the delays for all the messages received at the central personal area network device that were sent from the distributed server, and Figure 2 shows the jitters for the same messages, calculated as the difference between current and previous delay.

Figure 3 shows the delays for all the messages received at the distributed server that were sent from the central personal area device, and Figure 4 shows the jitters from the same messages.

From these figures we can see that the delay from the messages received at the central device of the PAN is always between 9 and 218 milliseconds. The Jitter from the same messages is always between 0 and 160 milliseconds.

We can also see that the delay from the messages received at the distributed server from the central device of the PAN is always between 7 and 215 milliseconds and the jitter varies from 0 to 134 milliseconds.

4.3 Testing communication between large scale distributed level servers

For testing communications between the large scale distributed level servers, we also implemented time stamping of messages the same way we implemented on the connection between the central device of the personal area network and a distributed server.

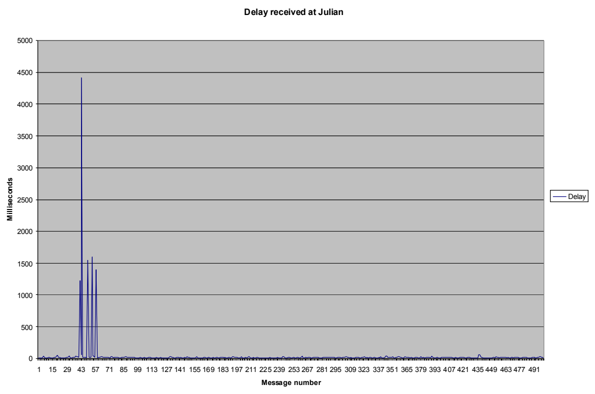

Figure 5 - Delays received from the distributed server Gillian on the distributed server Julian

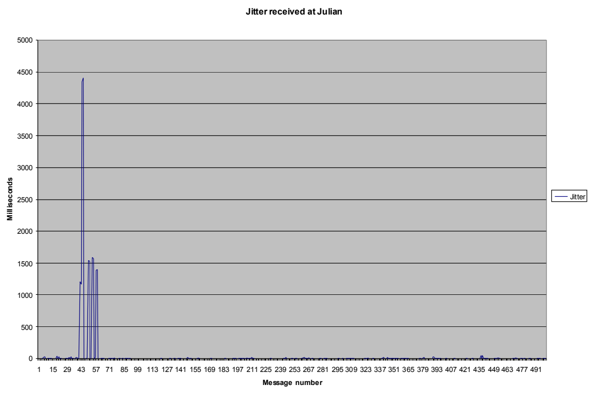

Figure 6 - Jitter received from the distributed server Gillian on the distributed server Julian

The message is marked with a timestamp equal to current time when sending and is marked again with the delay when receiving the message at the receiving end.

We also implemented logging of the received message delays (from which we can also derive the jitter), on the distributed servers.

Because we still do not have a concept application to run on the architecture, the only messages running on the distributed servers are the virtual time synchronization messages. And is the delay and jitter of these messages that's is shown on the graphics below.

So, Figure 5 represents the delays of all the messages transmitted from distributed server Gillian to distributed server Julian, while Figure 6 represents the jitters for the same messages.

Figure 7 represents the delays for messages received at distributed server Gillian transmitted from distributed server Julian, while Figure 8 represents the jitters for the same messages.

We only tested with two distributed servers, because that were sufficient for testing the effects of cryptography on communication and we are not full of resources.

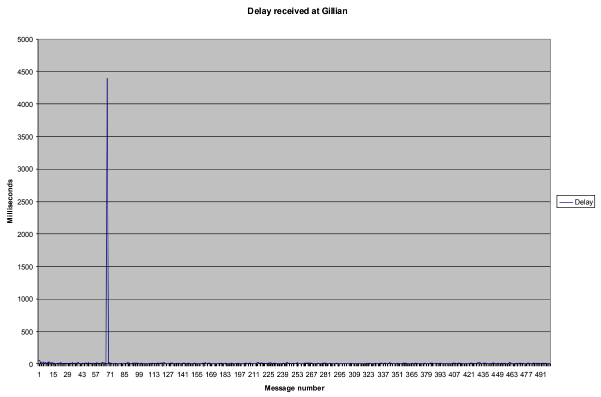

Figure 7 - Delay received from distributed server Julian on distributed server Gillian

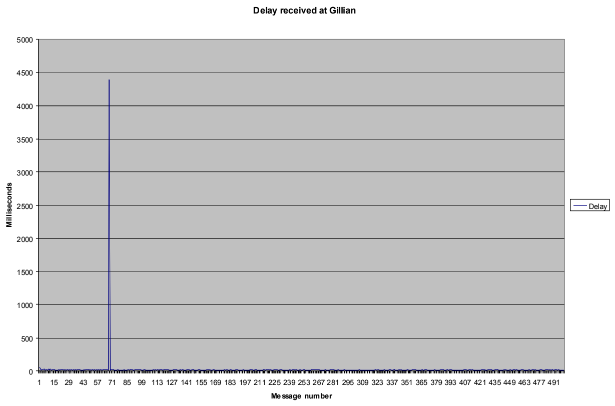

Figure 8 - Jitter received from distributed server Julian on distributed server Gillian

From these figures we can see that the delay from distributed server Gillian to distributed server Julian is always between 7 and 62 milliseconds, in fact most below 30 milliseconds, with the exception of some isolated values in the order of 4-5 seconds witch are obviously due to network loss or to processor overload or some other external factor. Jitter is normally between 0 and 50 (normally below 30), with the same isolated values.

Values for delay and jitter for the case where the transmission was from Julian tio Gillian are similar.

5. CONCLUSIONS

Through this work, we contributed with a privacy and security architcture for our middleware for large scale mobile and pervasive large scale mobile and pervasive augmented reality games.

We can conclude that our privacy and security architecture for our middleware for large scale mobile and pervasive augmented reality does its function well: It keeps data secure by encrypting it , it authorizes and authenticates users, servers and devices.

We also, through testing, proved that it has a acceptable amount of contribution to delay and jitter of the total architecture of the middleware, which, after applying security, continues running with acceptable values for delay and jitter for most interactive applications, including augmented reality.

Future work on this middleware platform will include QoS related work, Management related work, and proof-of-concept applications.

Acknowledgment

This work is being financed in part by the FCT – Portuguese Foundation for Science and Technology and FP6 CONTENT Network of Excelence (NoE).

REFERENCES

[1] Kimmo Raatikainen, Henrik Bærbak Christensen, Tatsuo Nakajima, “Application Requirements for Middleware for Mobile and Pervasive Systems”, Mobile Computing and Communications Review, Volume 6, Number 4, October 2002, pp. 16 – 24 , ACM Press

[2] Keith Mitchell, Duncan McCaffery, George Metaxas, Joe Finney, Stefan Schmid and Andrew Scott, “Six in the City: Introducing Real Tournament – A Mobile IPv6 Based Context-Aware Multiplayer Game”, Proceedings of NetGames'03, May 22-23, 2003, Redwood City, California, USA, pp. 91-100, ACM Press

[3] Hideyuki Tamura, Hiroyuki Yamamoto, and Akihiro Katayama, “Mixed Reality:Future Dreams Seen at the Border between Real and Virtual Worlds”, Virtual Reality, November/December 2001, pp. 64 –70, IEEE

[4] Nokia – Developer resources (Forum Nokia), http://www.forum.nokia.com/, Accessed April 2004

[5] Sony Ericsson Developer World, http://developer.sonyericsson.com/, Accessed April 2004

[6] Pedro Ferreira, “Network Middleware for Large Scale Mobile and Pervasive Augmented Reality Games” in Proc. of the CoNext 2005 - ACM Conference on Emerging Network Experiment and Technology, pp. 242-243, CoNext 2005 - ACM Conference on Emerging Network Experiment and Technology, Toulouse, France, October-2005

[7] Pedro Ferreira, João Orvalho, Fernando Boavida, ”Large Scale Mobile and Pervasive Augmented Reality Games”, in Proc. of the EUROCON 2005 - The International Conference on "Computer as a Tool", pp. 1775-1778, Vol. 1, # 1, EUROCON 2005 - The International Conference on "Computer as a Tool", Belgrade, Serbia and Montenegro, November-2005

[8] Pedro Ferreira, João Orvalho, Fernando Boavida, “A Middleware Architecture for Mobile and Pervasive Large Scale Augmented Reality Games”, in Proc. of the 5th Annual Communication Networks and Services Research Conference (CNSR 2007), pp. 203-212 , Vol. 1, #1, ISBN 0-7695-2835-X, IEEE Computer Society, 2007, Fredericton, New Brunswick, Canada

[9] 3GPP TS 33.102 v7.0.0, “3rd Generation Partenership Project; Technical Specification and System Aspects; 3G Security; Security architecture (Release 7)”, December 2005

[10] 3GPP – 3rd Generation Partnership Project , http://www.3gpp.org/

[11] Bluetooth Specifications, http://www.bluetooth.org/specs/

[12] Java APIs for Bluetooth (JSR-82), http://jcp.org/en/jsr/detail?id=82

[13] Pedro Ferreira, João Orvalho and Fernando Boavida ,”Sixrm: Full Mesh Reliable Source Ordered Multicast”, in Proc. of the SoftCom2006 - 14th International Conference on Software, Tellecommunications & Computer Networks, SoftCom2006 - 14th International Conference on Software, Tellecommunications & Computer Networks, Split, Croatia, September 2006

[14] T.Dierks, E.Reskorla, “The Transport Layer Security (TLS) Protocol version 1.1”, RFC 4346, IETF, April 2006

[15] S. Deering, R. Hinden, “Internet Protocol Version 6 (Ipv6) Specification”, RFC2460, IETF, December 1998

[16] João Gilberto de Matos Orvalho, “ARMS – Uma plataforma para aplicações multimédia distribuídas, com qualidade de serviço”, Phd Thesis, December 2000, DEI-FCTUC

[17] João Orvalho, Fernando Boavida, “Augmented Reliable Multicast CORBA Event Service (ARMS): a QoS-Adaptive Middleware”, in Lecture Notes in Computer Science, Vol. 1905: Hans Scholten, Marten J. van Sinderen (editors), Interactive Distributed Multimedia Systems and Telecommunication Services, Springer-Verlag, Berlin Heidelberg, 2000, pp. 144-157. (Proceedings of IDMS 2000 – 7th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services, CTIT / University of Twente, Enschede, The Netherlands, October 17-20, 2000).

[18] Pedro Ferreira, João Orvalho and Fernando Boavida, “Middleware for embedded sensors and actuators in mobile pervasive augmented reality”, in Proc. of the INFOCOM 2006 (IEEE XPLORE), INFOCOM 2006 Student Workshop, Barcelona, April 2006